X 魔方应用开发指南

目　录

1 编程向导.....4
应用概述.....4
项目.....4
使用项目创建应用程序.....5
使用应用程序向导创建项目.....5
项目编辑.....6
项目调试.....6
资源.....6
图像资源.....7
界面资源.....7
物品资源.....8
模型资源.....8
材质资源.....8
骨架资源.....8
动画集资源.....8
声音资源.....9
如何使用资源.....9
脚本.....10
Lua 脚本语言简介.....10
脚本分类.....10
项目脚本.....10
窗口脚本.....10
场景脚本.....11
物体脚本.....11
普通脚本.....11
脚本扩展.....11
Lua 导出类.....11
Lua 导出函数.....12
Lua 事件.....12
脚本编辑与调试.....16
示例 I'm a Tree.....16
2 编辑与调试.....19
2.1 项目文件概述.....19
2.2 XMaker 工具概述.....19
2.3 Hello,World 示例.....19
2.4 编辑.....23
2.4.1 项目编辑.....23
项目的创建/添加.....23
相关文件的添加.....24

2.4.2 场景编辑.....25
2.4.3 界面编辑.....26
窗口控件类型.....26
窗口的创建.....26
窗口的销毁.....32
窗口属性和行为.....32
窗口消息与交互.....32
窗口示例.....36
2.4.4 脚本编辑.....41
Lua 代码编写.....42
Lua 代码调试运行.....42
2.4.5 物品编辑.....42
2.4.6 图像编辑.....42
2.4.7 模型编辑.....43
2.4.8 骨架编辑.....43
2.4.9 动画编辑.....43
2.4.10 材质编辑.....43
2.5 调试.....43
调试工具栏.....44
输出窗口.....44
调用堆栈.....45
断点窗口.....45
局部变量窗口.....46
监视窗口.....46
3 开发 API.....47
3.1 概述.....47
3.2 宏.....47
如何定义宏.....47
3.3 内置对象与内置函数.....48
内置对象.....48
__MainObject.....48
内置函数.....50
功能描述.....50
函数列表.....50
3.4 常用对象.....53
3.4.1 基本对象.....53
LXPoint.....53
LXRect.....58
LXRgn.....80
LXColor.....90
LXFont.....98
LXHandle.....110
LXPublic.....113
LXImageList.....127

LXCursor..... 131
3.4.2 图像与声音.....135
LX2Dshape..... 135
LXTexture..... 150
LXImage..... 174
LXSound..... 181
3.4.3 网络功能.....185
网络开发简述.....185
LXNetServer..... 185
LXNetClient.....188
3.5 窗口 API..... 192
LXWnd.....192
LXButton..... 247
LXStatic..... 248
LXEdit.....250
LXSlider.....260
LXProgress..... 264
LXListCtrl.....268
LXHeadCtrl.....286
LXComboBox.....296
LXListBox.....311
LXMenu.....327
LXGroupBox..... 337
LXTabCtrl..... 340
LXToolTip.....347
LXTreeCtrl.....351
LXDialog..... 381
LXColorDialog..... 384
LXFileDialog..... 387
LXFontDialog..... 392
消息对话框.....398
3.6 场景 API..... 400
3.6.1 基本对象.....401
LXAABBBox..... 401
LXAABBVolume.....424
LXBoundBox.....431
LXMatrix..... 440
LXPlane..... 462
LXQuaternion..... 471
LXVector2.....487
LXVector3.....505
LXVector4.....527
3.6.2 场景对象.....546
LXSceneCamera..... 546

LXSceneManager..... 560
LXSceneItem..... 571
LXSceneNode.....618
LXParticle.....654
3.7 事件.....657
3.8 高级应用模块.....657
LXChatWnd.....657
功能描述.....657
使用事项.....658
函数列表.....658
4 应用示例.....672
示例一.....672
示例二.....672
示例三.....672
示例四.....675
其它示例.....675

1 编程向导

应用概述

每一个 X 魔方应用，对应着一个或者多个项目文件（XSN 文件）。

根据展现形式，分为以下几类：
安装到 X 魔方的应用：此类应用的展现形式是嵌入到 X 魔方网页或者 X 魔方客户端中。
桌面程序：此类应用不会展现在 X 魔方网页或者 X 魔方客户端中，是一个桌面程序。

根据项目类型，分为以下几类：
普通 Lua 项目：Lua 工程
2D 图像项目：2D 图像工程
3D 场景项目：3D 场景工程

制作一个 X 魔方应用，实质上就是对项目文件及其所属文件生成与编辑，然后经过调试得到正确的结果。

项目

项目是一组文件。项目构成应用。

同时，项目是解决方案的单独部分，用于解决不同的应用程序要求。解决方案定义包括项目

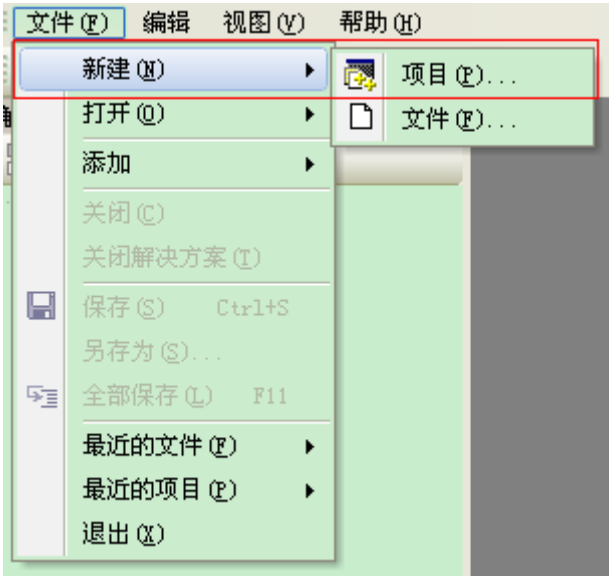
间的依赖项关系。它们也包括有关如何部署构成解决方案的项目的信息。

使用项目创建应用程序

创建新项目的最好方法是使用 X 魔方应用程序向导。有关详细信息，请参见用使用应用程序向导创建项目。

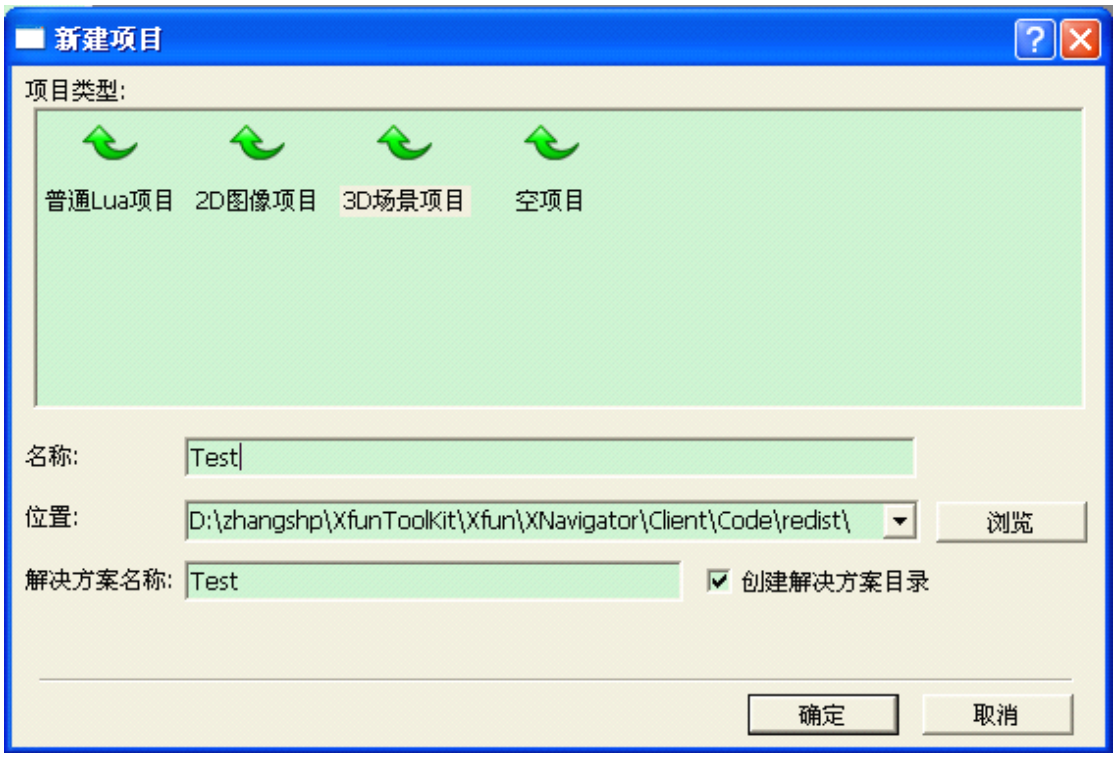
始终可以不使用应用程序向导编写程序，但是因为向导提供一组完整的项目文件和结构，所以使用向导开始项目使您得以立即编写程序。若要创建空项目，请使用项目向导。可以将自己的文件添加到此项目。此方法适合简单的应用程序。

使用应用程序向导创建项目



用户可以使用 X 魔方编辑器的新建项目向导来创建自己的项目，用户可以点击“文件”菜单中的“新建”，再单击“项目”选项

或者直接点击工具栏中的新建项目按钮都可以弹出新建项目向导



选择自己需要的项目类型，项目存放目录，然后输入项目名称，点击确定，就可以创建好一个项目。

项目编辑

项目编辑是整个应用开发的核心。它主要包含两个部分：一是项目相关资源文件的添加与编辑，这些构成应用的显示元素；另外一部分是脚本的编辑。脚本作为一种特殊的项目文件，它控制应用程序的流程，逻辑与部分数据。

详细内容请参考编辑。

项目调试

项目的调试主要是指对脚本的调试。

详细内容请参考调试。

资源

项目由各种资源文件构成，每种类型的资源又可以有多个文件构成。X 魔方应用中的主要资源可以分为图像资源、界面资源、物品资源、声音资源。

其中，界面资源可以包含图像资源，声音资源。

物品资源可以包含模型资源，材质资源，骨架资源，动画集资源。物品资源主要应用于

3D 场景项目中。

图像资源

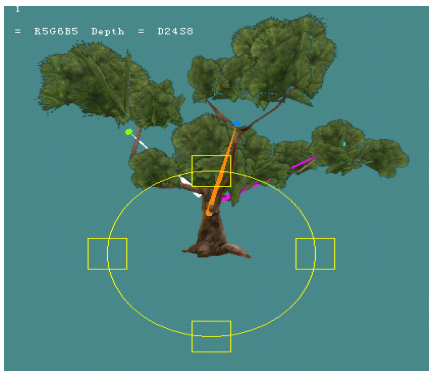
用户可以在自己的项目中添加常用的图片文件做为图像资源，比如 xmp,jpg,bmp 等。这些图像资源可以用来作为材质或者界面的一部分，也可以直接在场景中渲染。

界面资源

界面资源以对话框的形式呈现的用户面前。一个最简单的界面如上图所示。
用户可以使用 XMaker 来创建和编辑各种各样独具特色的界面资源，例如可以向界面资源中添加图像资源来改变窗口或者控件的背景图，可以向对话框中添加脚本来响应界面中的各种事件。

详细内容请参考界面编辑。

物品资源

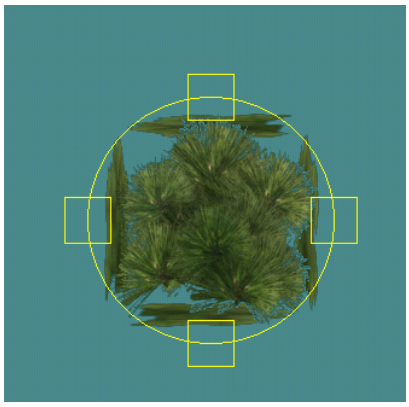


物品是一种综合资源，它由一个模型文件、骨架、动画和材质组合而成，呈现在用户面前的是一个动态 3 维立体模型。应用于 3D 场景项目中。

模型资源

模型资源是指一个 3 维物体的模型文件。应用于 3D 场景项目中。

材质资源



物品的材质决定了它能反射什么颜色的光线以及反射多少光线，对于用户来说就是用于区分物品的质地。应用于 3D 场景项目中。

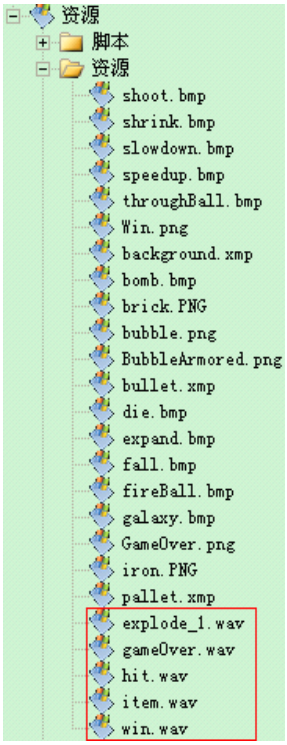
骨架资源

骨架资源是支撑模型实现模型动画的基础。应用于 3D 场景项目中。

动画集资源

动画集资源是指骨架动画集，用户可以设置一个动画集，用来控制模型骨架的运动方式。应用于 3D 场景项目中。

声音资源



用户可以选择 WAV 或 Mp3 声音文件加入到应用项目中做为声音资源。

如何使用资源

资源创建或者添加成功之后，用户可以调用相应的 API 函数去获得指定的对象，然后进行相关操作。

举例说明：

```
--XMain.lua 文件
#include “XMain.h”

loadlib(“XLualib”)
loadlib(“XLuaBase”)
loadlib(“XGraphBase”)
loadlib(“XXRCCreator”)

X_ON_LOADLUA(OnMainLuaLoad)

function OnMainLuaLoad(this, nEventID, SendName, param)
    --创建对话框对象
dialog = LXDialog.new(“..\\界面\\Dialog.xrc”)
--创建非模态对话框
```

```
dialog.Create()
--显示对话框
dialog.ShowWindow()
end
```

上面的示例中中，Dialog.xrc 是通过 X 魔方编辑工具创建的对话框资源文件，通过调用对话框导出类的 new 函数创建对话框对象，然后调用 Create 函数创建一个非模态的对话框，最后调用 ShowWindow 函数将其显示出来。

上面示例中的代码详细内容请参考脚本。

脚本

资源组成了 X 魔方应用的物质世界，脚本则是整个 X 魔方应用的核心。用户可以通过编写脚本实现流程控制，应用逻辑以及数据处理。

X 魔方应用采用 Lua 脚本语言。与其它类型的脚本相比较，Lua 语言具有简洁，轻量，可扩展，高效的特性。

Lua 脚本语言简介

Lua 语言是一门简洁、轻量、可扩展的脚本语言。Lua 是一种动态类型语言，没有类型的定义，编码时无需声明变量类型。
Lua 语言中有 8 种基本类型的变量：nil, boolean, number, string, userdata, function, thread 和 table。
具体内容参考 http://www.lua.org/manual/5.1/manual.html。

脚本分类

根据脚本的所属对象，可以分为五类：

项目脚本

一般来说，每个项目都至少包含一个项目脚本，其中有一个为项目主脚本，默认情况下为 XMain.lua。它是项目启动的主 Lua 脚本。

窗口脚本

一般来说，每个对话框都可以关联一个 Lua 脚本。
它负责窗口与其它对象之间的交互。
详细内容请参考窗口消息与交互和窗口示例。

场景脚本

创建 3D 场景项目会默认一个场景脚本(Scene.lua)。
它负责场景与其它对象之间的交互。

物体脚本

物体脚本负责物体与其它对象之间的交互。

普通脚本

除上述脚本外，其它脚本均可视作普通脚本。它与上述脚本一起构成了应用的脚本系统。

脚本扩展

X 魔方应用扩展了标准 Lua 语言。它向用户提供了一套 API，例如提供了图像的绘制，声音的播放，窗口或场景的创建等。

其扩展主要体现为以下三种方式：
1 导出类
2 导出函数
3 事件

Lua 导出类

Lua 导出类是由 X 魔方编辑器提供的一组 Lua 封装类，导出类由成员函数和成员变量组成。用户可以根据自身需要使用不同的导出类。

创建导出类对象

使用导出类之前要先创建导出类对象，这里有两种创建方法供用户选择：
第一，通过”.”操作符调用相应的 Lua 导出类中的 new 函数创建导出类对象，比如:

```
point = LXPoint.new()                --通过 new 方法创建一个导出类对象
```

第二，通过某个函数返回一个 Lua 导出类的对象，比如

```
point = otherObj.GetPoint()          --otherObj.GetPoint 函数的返回值类型为 LXPoint
```

使用导出类对象

导出类对象创建之后，可以通过”.”操作符调用导出类的所有成员函数或者访问成员变量

例如：

```
obj.variant = 8                    --obj 对象的 variant 变量其值设为 8
point.GetX()                       --得到 point 对象的 X 坐标
print(rect.left)                   --输出 rect 对象的左上角 X 坐标
```

Lua 导出函数

Lua 导出函数是由 X 魔方编辑器提供的一组函数，它们不属于 Lua 导出类的成员，可以在用户的 Lua 脚本中直接调用，它们的返回值可能是导出类对象，也可能没有返回值。
例如：

```
--调用导出函数 print 打印消息
print("Lua Load")
```

详细内容请参考内置对象与内置函数。

Lua 事件

事件是一系列消息，事件机制给整个系统提供了灵活的交互方式。

事件主要由四部分组成：

事件 ID	nEventID	唯一标识一个事件
事件发生方	SendName	发生事件的对象名称
事件参数	param	当事件发生时对应事件的参数
事件响应函数	function	当事件发生时处理函数

当某种类型的事件发生时，系统会发送附带指定参数的消息并触发对应的事件响应函数。

用户也可以自定义事件来捕获指定的消息。请参考如何使用自定义事件。

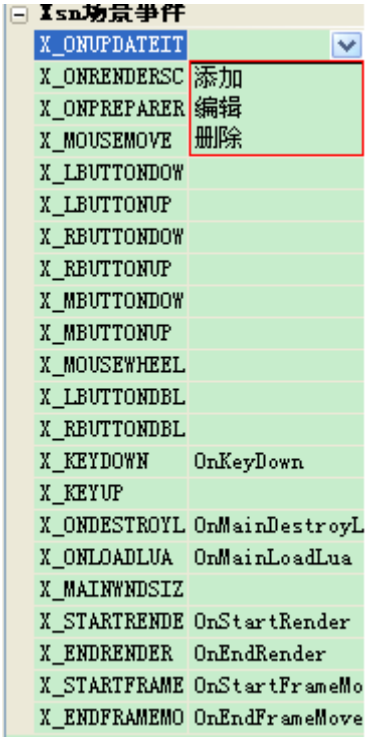
事件类型

事件可以分为：系统事件(基本事件、场景事件、窗口/控件事件、网络事件)、用户自定义事件。

脚本中的宏

参见宏。

如何自动添加事件



事件分为系统事件和用户自定义事件，其中系统事件可以使用 X 魔方编辑器自动添加，用户也可以手动添加，而用户自定义事件则只能通过手动添加。
场景或窗口都具有自己的属性，它们会显示在 X 魔方编辑器右侧的属性栏里，位于最下方的就是场景或窗口可以响应的事件，单击事件名可以看到事件说明，单击后方的 ComboBox 控件会弹出下拉框，用户可以选择添加、编辑或是删除已经添加的事件。

如何手动添加事件

手动添加事件需要用户手动添加脚本代码。

系统事件的手动添加分为如下两步:
第一步，通过项目的 XMain.h 文件查找需要响应的事件及其响应宏的名称。
第二步，在需要响应该事件的 Lua 脚本中加入事件响应宏，编写响应函数。

用户也可以自定义事件。

如何响应事件

用户可以使用 X 魔方编辑器的内部宏来定义消息响应函数，示例如下：

```
#include “XMain.h”

loadlib(“XLualib”)

X_ON_MOUSEMOVE(OnMouseMove)

function     OnMouseMove(this, nEventID, SendName, param)
        local x = param.x
        local y = param.y
        print(x)
        print(y)              --打印当前鼠标的 X、Y 坐标
end
```

上例所示为鼠标移动消息的消息响应，用户在自己的 Lua 文件中加入 X_ON_MOUSEMOVE(function)宏，其中 function 为响应函数的字符串名称，当窗口或者场景发生鼠标移动事件时，系统会自动调用以 function 命名的这个函数，并传入消息指定的参数。

消息响应宏及参数内容请参考附录中的 XMain.h。

如何自定义事件

用户自定义事件的添加一可以分为如下两步
第一步，在触发方调用接收事件对象的 FireEvent 函数，指定事件 ID，产生对象，事件参数。
第二步，在接收事件对象的 Lua 脚本中通过使用 ON_XEVENT 宏，指定对应的事件 ID，产生对象，响应函数，并编写响应函数内容。

注意用户自定义事件的 ID 取值为 10000 之后的值。

示例如下：

```
--Dialog1.lua  第一个对话框脚本
#include “XMain.h”
loadlib(“XLualib”)
loadlib(“XLuaBase”)
loadlib(“XGraphBase”)
loadlib(“XXRCCreator”)

X_ON_LOADLUA(OnMainLuaLoad)

function OnMainLuaLoad(this, nEventID, SendName, param)
      --创建第二个对话框
Dialog2 = LXDialog.new(“..\\界面\\Dialog2.xrc”)
--Dialog1 触发 Dialog2 脚本中的事件
Dialog2.FireEvent(10001, “Dialog1”, 100)
```

```
end
```

```
--Dialog2.lua  第二个对话框脚本
#include “XMain.h”

loadlib(“XLualib”)
loadlib(“XLuaBase”)
loadlib(“XGraphBase”)
loadlib(“XXRCCreator”)

--定义 Dialog1 通知事件的响应函数
ON_XEVENT(10001, “Dialog1”, “OnDialog1Notify”)

function      OnDialog1Notify(this, nEventID, SendName, param)
--判断参数是 100，则打印 Hello World
if param == 100 then
      print(“Hello World”)
end
end
```

系统事件

基本事件：

X_ONLOADLUA	当加载 Lua 对象时触发
X_ONDESTROYLUA	当摧毁 Lua 对象时触发
X_MAINWNDSIZE	主窗口大小改变事件
X_STARTRENDER	当开始渲染循环时调用
X_ENDRENDER	当结束渲染循环时调用
X_STARTFRAMEMOVE	当开始更新状态时调用
X_ENDFRAMEMOVE	当结束更新状态时调用

场景事件：

X_ONPREPARENDER	当开始准备渲染场景时触发
X_ONUPDATEITEM	当更新场景物品状态时触发
X_ONRENDERSCENE	当开始绘制场景时触发

窗口事件：

X_ONWINDOWSIZE	当窗口改变大小时触发
X_ONWINDOWMOVE	当窗口改变位置时触发
X_ONWINDOWSHOW	当窗口显示方式发生改变时触发
X_ONSCROLLCHANGED	当窗口发生滚动时触发
X_ONDIALOGINIT	当对话框初始化时触发

X_ONDIALOGDESTROY	当对话框销毁时触发

控件事件：

X_ONEDITENTER	当 Edit 控件回车时触发
X_ONBUTTONCLICK	当 Button 按钮点击时触发
X_ONCOMBOBOXSELECT	当 ComboBox 选中项时触发
X_ONLISTBOXSELECT	当 ListBox 选中项时触发
X_ONMENUSELECT	当 Menu 选中项时触发
X_ONTABCTRLSELECT	当 TabCtrl 选中项时触发
X_ONLISTCTRLSELECT	当 ListCtrl 选中项时触发
X_ONLISTCOLUMNCLICK	当 ListCtrl 选择列时触发
X_ONSTATICCLICK	当 Static 点击时触发

网络事件：

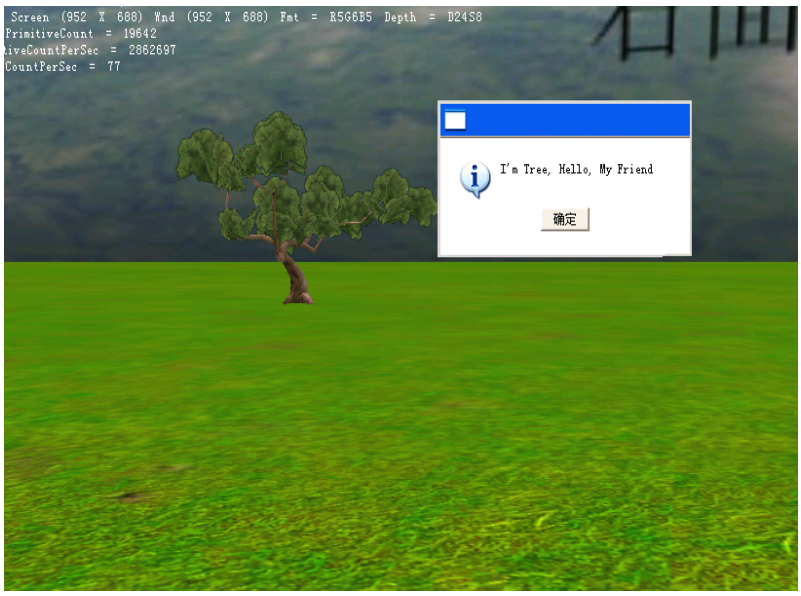
X_ONRECEICEMSG	当成功接收到网络消息时调用
X_ONNEWCONNECT	当服务器接收到新的连接时
X_ONCONNECTBREAK	当连接断开时

消息响应宏及参数内容请参考附录中的 XMain.h。

脚本编辑与调试

详细内容参考脚本编辑。

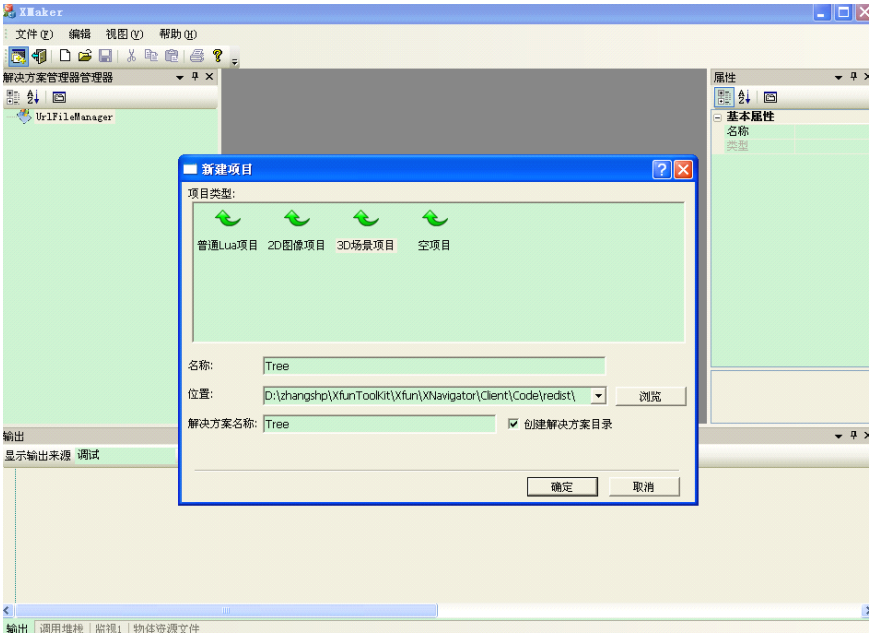
示例 I'm a Tree



这是一个简单的应用，包含一个物体资源（一棵树），当用户点击这个物体时，会弹出一个

MessgeBox 向用户打招呼。下面介绍一下这个应用的实现过程:

第一步，通过 XMaker 的向导创建名为”Tree”的 3D 场景项目：



第二步，将树的模型文件、骨架文件、材质文件、动画集文件加入项目做为资源：

第三步，通过编写脚本来显示树，打开场景的主 Lua(这里为 Scene.lua 文件)，加入以下代码：

```
--加入鼠标左键点击事件
X_ON_LBUTTONDOWN(OnLButtonDown)
```

```
--在系统第一次加载脚本时创建物体对象
function   OnLoadLua(this,nEventID,SendName,param)
      --使用场景管理器加载导入的资源文件，它会返回一个对象，通过它就可以操作这
--个物体对象了,这时就可以在屏幕上看到一颗松树了
      Item = this.CreateSceneItem("Item", "..\\资源\\黄山松.xmo", "..\\资源\\黄山松.xbo")--
end
```

第四步，向脚本中添加事件响应：判断用户是否选中该松树，如果选中则弹出消息对话框向用户打招呼，如下图单击事件添加按钮，

X 魔方编辑器会自动在代码中加入鼠标左键单击事件的消息响应宏 X_ON_LBUTTONDOWN(OnLButtonDown)，
并生成一个响应函数 function　OnLButtonDown(this,nEventID,SendName,param)，在该函数中添加如下代码

```
--鼠标按下的消息处理
function   OnLButtonDown(this,nEventID,SendName,param)
      --这里的 this 表示场景管理器的对象，通过它可以得到当前场景的相机
      camera = this.GetSceneCamera()
      --获得鼠标按钮的位置
      local Point = LXPoint.new(param.x, param.y)
      --将鼠标点转化为三维空间的一条射线
      local vecOri, vecDir = camera.GetRayFromScreenPt(Point)
      --然后判断这条射线是否与物体发生了碰撞
      local Res = Item.TestInterSectDirection(vecOri, vecDir, 10000, "ace")
      --如果碰撞说明选中了物体，弹出提示
      if Res.Distance > 0 then
```

```
            XMessageBox("I'm Tree, Hello, My Friend")
        end
end
```

到这里一个简单的应用就做好了，赶快运行一下看看效果吧~~~

2 编辑与调试

2.1 项目文件概述

项目文件(XSN 文件)格式与 html 类似，采用 XML 格式，以纯文本方式保存。

项目文件的生成与编辑采用 XMaker 工具。

项目所属文件主要由界面文件，物品文件，和脚本文件等各种资源文件组成。

在项目中用户可以新建各种类型的文件，或者添加已有的文件。当文件比较多时，可以通过新建筛选器来进行组织分类。

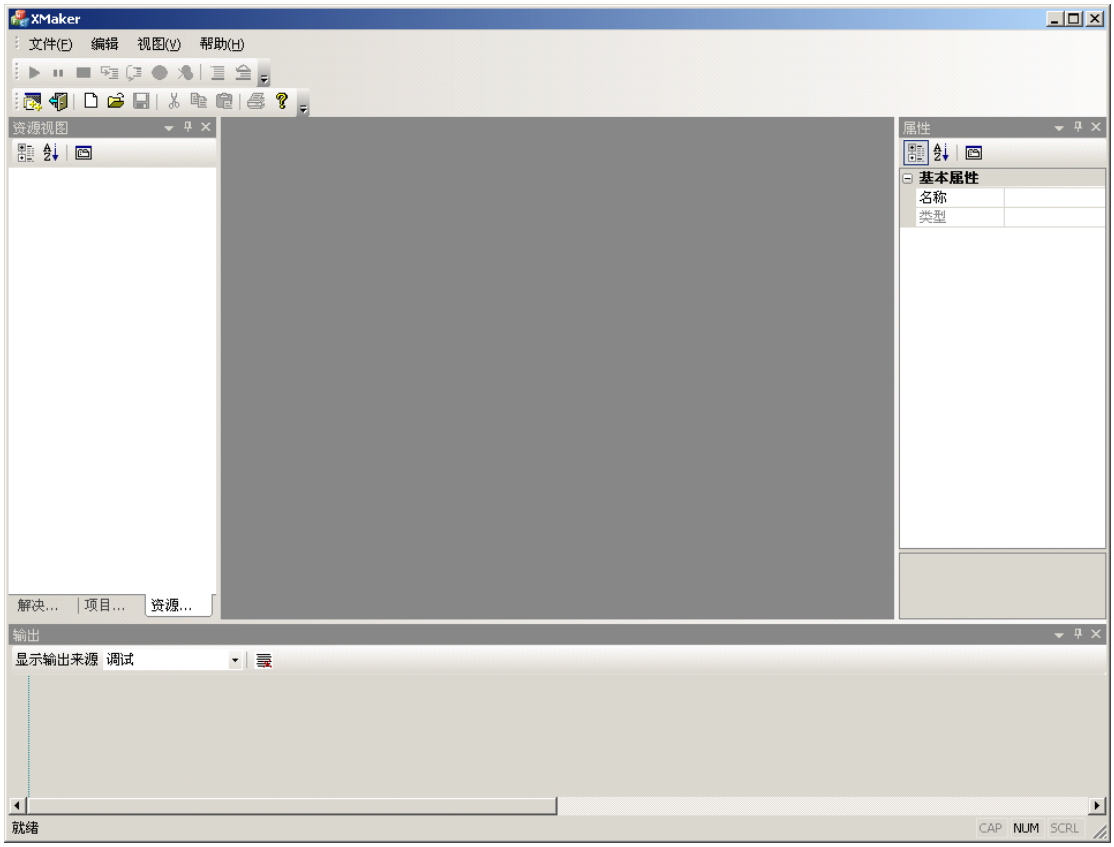
界面文件和物品文件及其它各种资源文件构成了 X 魔方应用的显示元素，而应用的部分数据与逻辑控制是通过脚本文件来进行的。

2.2 XMaker 工具概述

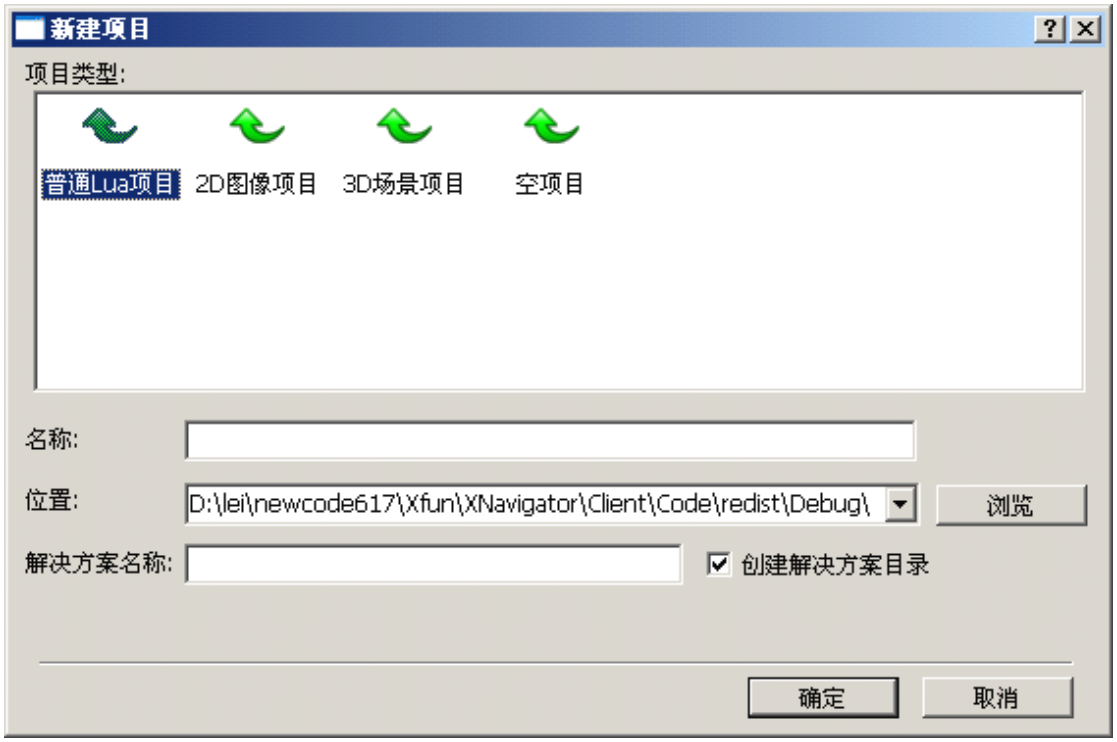
打开 XMaker.exe, 会出现如下图所示的一个窗口：
默认情况下，此窗口包含以下几个部分：
1 菜单
2 工具栏
3 视图：如解决方案视图，项目资源管理视图，资源属性，属性栏，输出等视图。用户可以通过菜单栏中的视图项来控制视图的显示/关闭，通过拖拽来控制视图的显示位置等。

2.3 Hello,World 示例

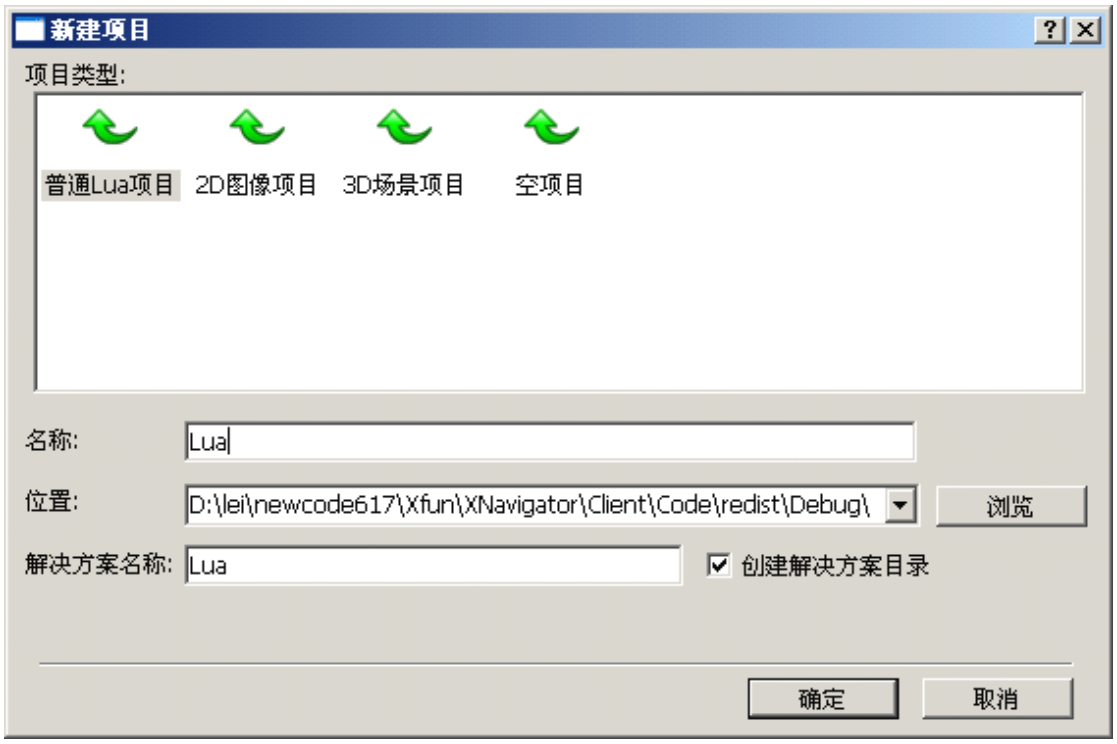
此示例将演示新建一个项目，并在输出的调试窗口中打印一个字符串”HelloWorld”。
具体步骤如下:
1 打开 XMaker.exe，出现如图所示界面：



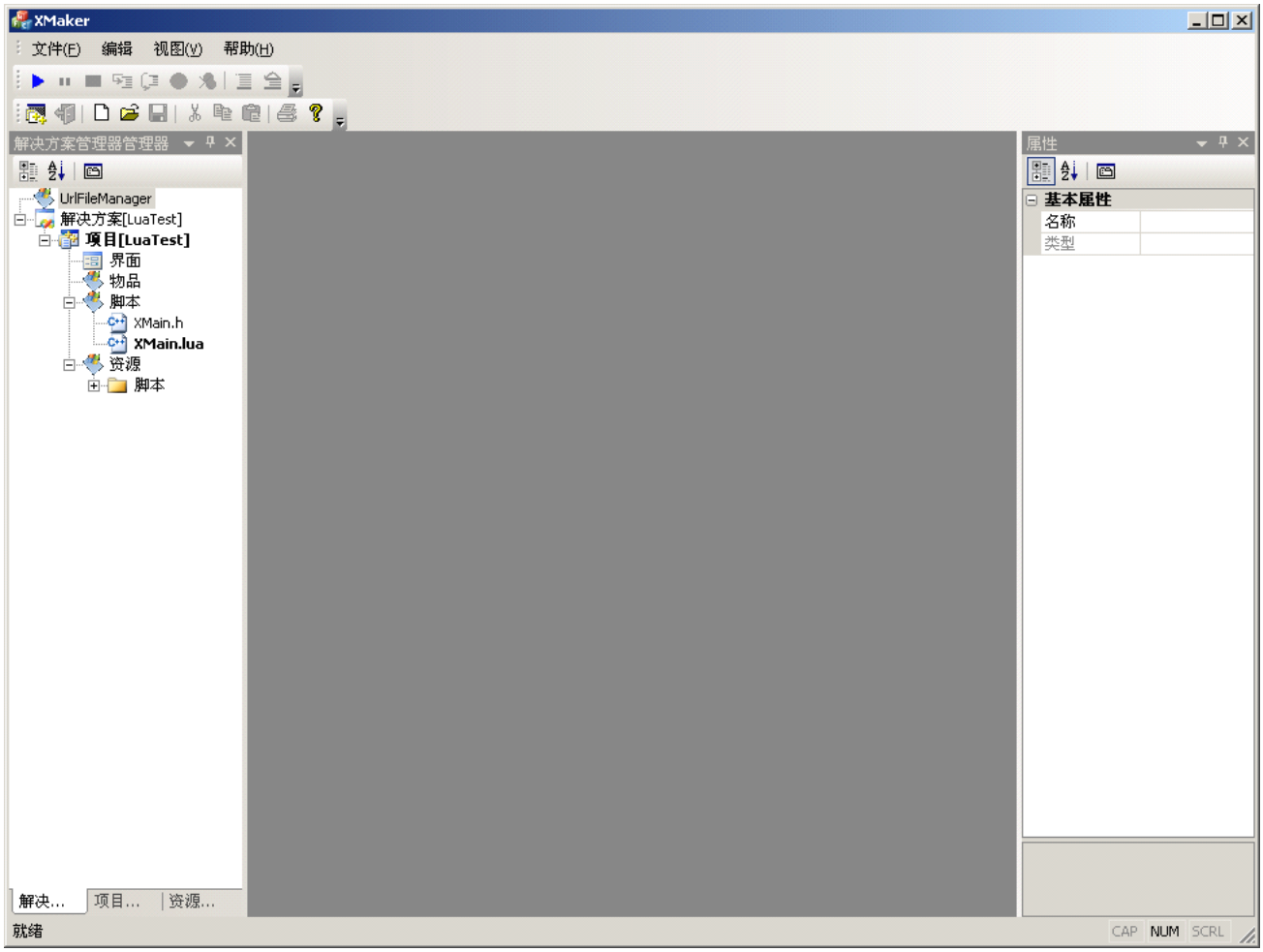
2 点击工具栏的“新建项目文件”按钮，弹出如下对话框。



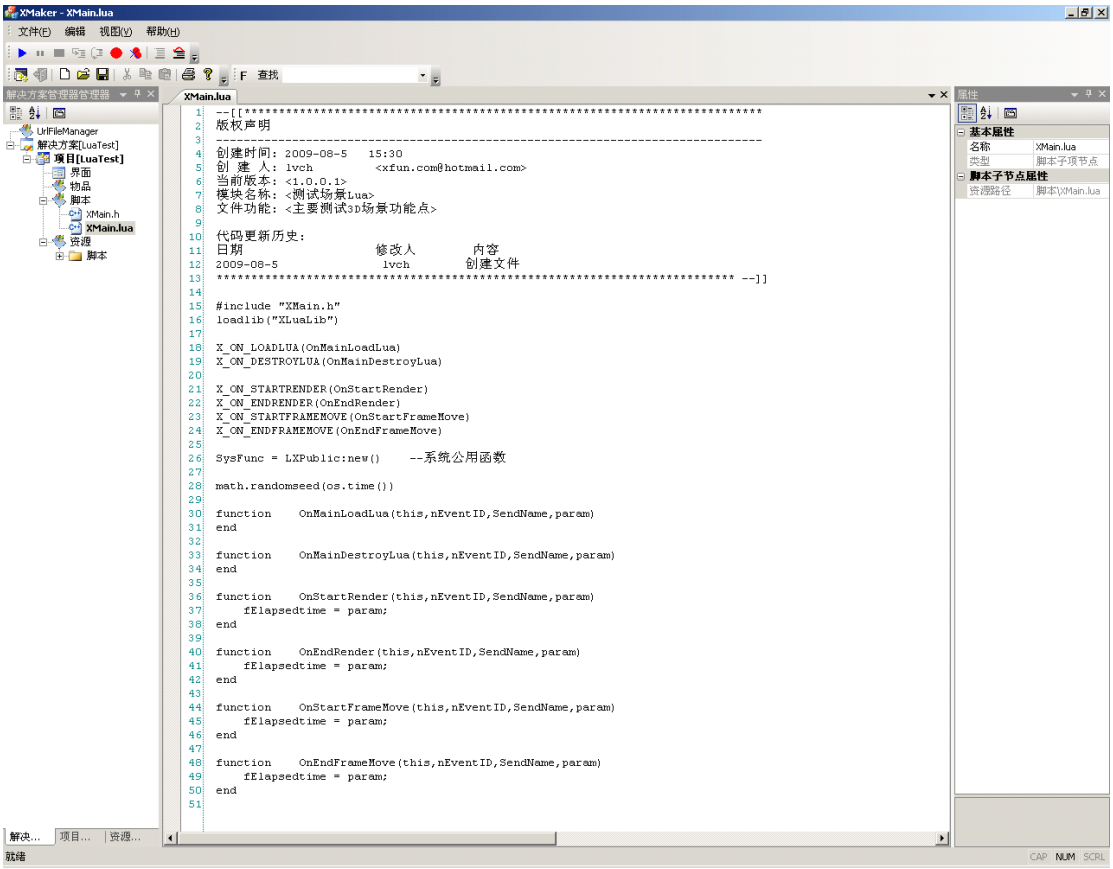
3 如下，在“项目类型”选项中选择“普通 Lua 项目”。在“名称”中写上名称，点击“浏览”按钮选择文件目录，在“解决方案名称”中可更改名称。点击确定按钮。



4 出现如下界面。双击左边视图的“XMain.lua”节点。

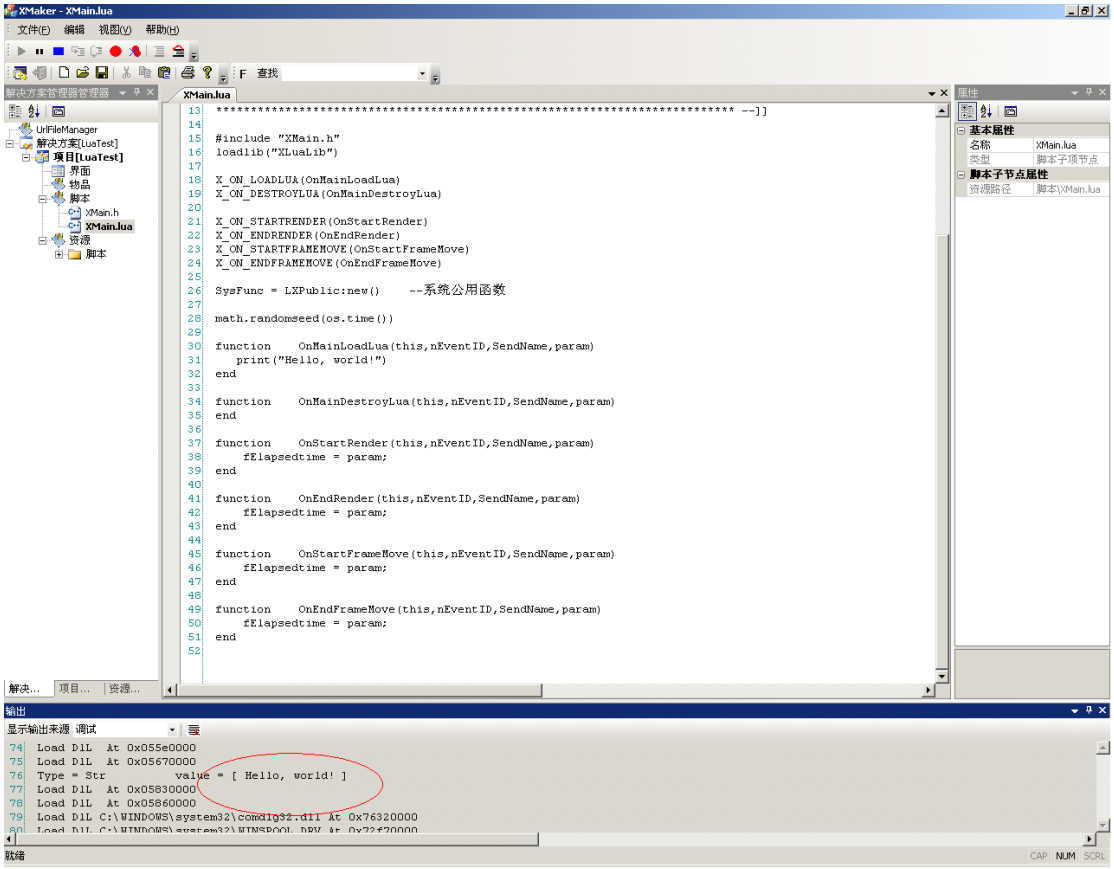


5 出现如下界面：



在 XMain.lua 中找到语句“function OnMainLoadLua(this,nEventID,SendName,param)”，在其后的一行输入语句
“print(“Hello, world!”)”.点击保存按钮。

6 按下 F5 键，此时将启动调试程序进行调试。同时打开视图-》输出视图， 即可发现输出语句“Hello, world!”.如下图所示：




```
--[[********************************************************************************

#include "XMain.h"
loadlib("XLuaLib")

X_ON_LOADLUA(OnMainLoadLua)
X_ON_DESTROYLUA(OnMainDestroyLua)

X_ON_STARTRENDER(OnStartRender)
X_ON_ENDRENDER(OnEndRender)
X_ON_STARTFRAMEMOVE(OnStartFrameMove)
X_ON_ENDFRAMEMOVE(OnEndFrameMove)

SysFunc = LXPublic:new()     --系统公用函数

math.randomseed(os.time())

function     OnMainLoadLua(this,nEventID,SendName,param)
   print("Hello, world!")
end

function     OnMainDestroyLua(this,nEventID,SendName,param)
end

function     OnStartRender(this,nEventID,SendName,param)
    fElapsedtime = param;
end

function     OnEndRender(this,nEventID,SendName,param)
    fElapsedtime = param;
end

function     OnStartFrameMove(this,nEventID,SendName,param)
    fElapsedtime = param;
end

function     OnEndFrameMove(this,nEventID,SendName,param)
    fElapsedtime = param;
end
```

2.4 编辑

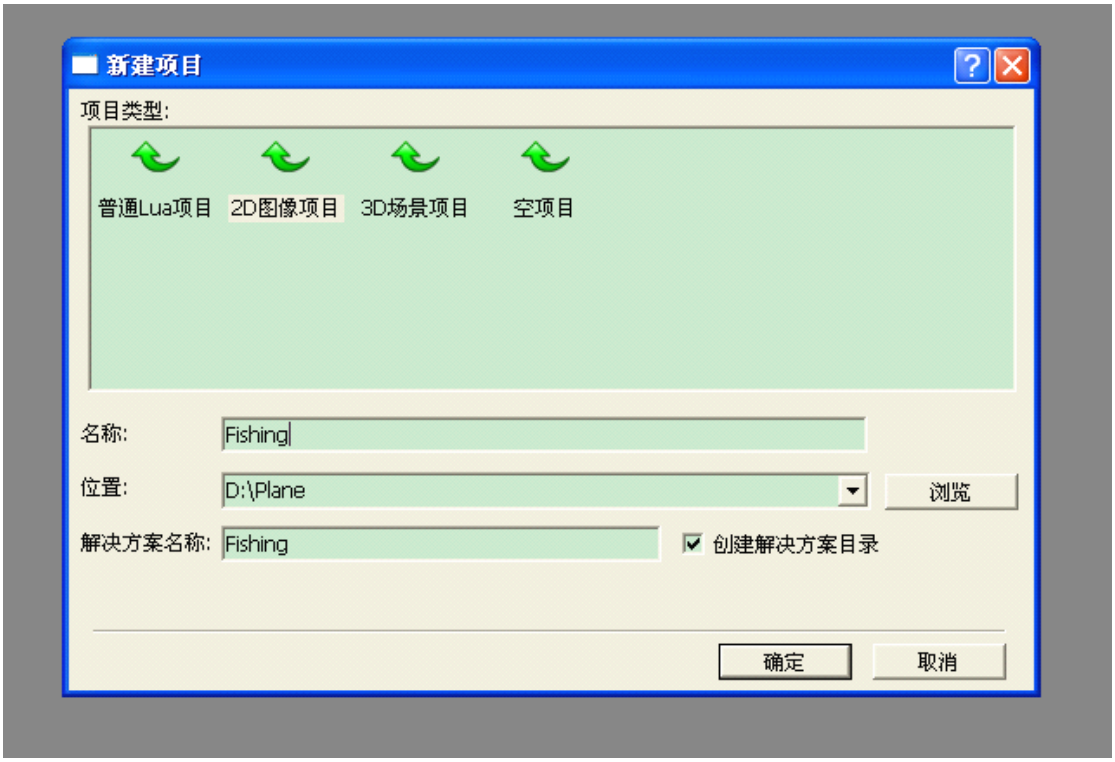
每一个 X 魔方应用都对应着一个 XSN 或者多个 XSN 文件。编辑都是围绕 XSN 文件及其相关资源进行的。
概括地讲，项目编辑建立应用的环境；场景编辑和界面编辑建立起应用的元素；脚本编辑控制应用的流程和逻辑，而其它编辑均可视作是辅助性的编辑。

2.4.1 项目编辑

项目编辑整个项目及各项资源的创建。

项目的创建/添加

启动 XMaker 后,点击文件->新建->项目 后将出现新建项目对话框,如下图:



根据需要创建某种类型（其具体含义参见 X 魔方应用概述）的项目。

也可以将一个现有的项目添加到一个现有的解决方案中。

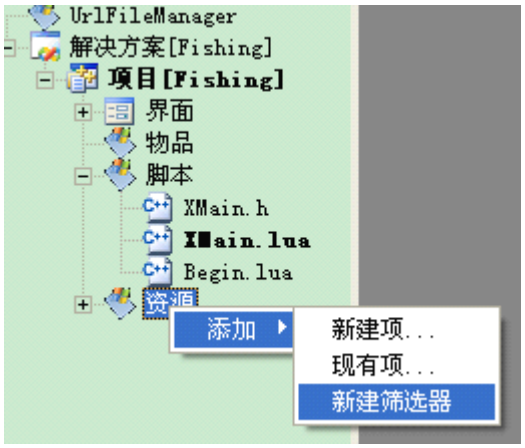
项目工程创建后，会产生一些默认的项和文件（空项目除外）。
这些默认项包括：
XMain.h　宏定义文件
XMain.lua　项目的主 Lua 文件：项目运行时将加载此 Lua 开始运行。

项目下包含界面　物品　脚本　资源节点　脚本下包含 XMain.h 和 XMain.lua

相关文件的添加

添加->新建项,将新建指定的文件
添加->现有项,将用户选择的文件添加到指定的资源目录下
添加->新建筛选器，将新建一个目录节点

一般来说，界面及相关资源一般在界面项中添加；物品及相关资源在物品项中添加；脚本及相关资源在脚本项中添加。



2.4.2 场景编辑

场景编辑负责物品，花草，天空体，地形和地貌的编辑。针对 3D 场景项目。

2.4.3 界面编辑

界面编辑主要是可视化产生 X 魔方应用中的窗口和控件。

窗口控件类型

X 魔方的窗口系统由以下类型的窗口或控件组成：

类型	含义	示例
LXWnd	基本窗口	Samples/Test_Wnd/ Test_LXWnd
LXButton	按钮	Samples/Test_Wnd/Test_Ctrls/ Test_LXButton
LXComboBox	组合框	Samples/Test_Wnd/Test_Ctrls/ Test_LXComboBox
LXEdit	编辑框	Samples/Test_Wnd/Test_Ctrls/ Test_LXEdit
LXGroupBox	分组框	Samples/Test_Wnd/Test_Ctrls/ Test_LXGroupBox
LXHeadCtrl	列表头控件	Samples/Test_Wnd/Test_ListCtrl/Test_LXHeadCtrl
LXListBox	列表框	Samples/Test_Wnd/Test_ComoboBox/LXListBox
LXListCtrl	列表	Samples/Test_Wnd/Test_ListCtrl/Test_LXListCtrl
LXMenu	菜单	Samples/Test_Wnd/Test_ComoboBox/LXMenu
LXProgress	进度条	Samples/Test_Wnd/Test_Ctrls/ Test_LXProgress
LXSliderCtrl	滑动条	Samples/Test_Wnd/Test_Ctrls/ Test_LXSliderCtrl
LXStatic	静态文本	Samples/Test_Wnd/Test_Ctrls/ Test_LXStatic
LXTabCtrl	标签页	Samples/Test_Wnd/Test_Ctrls/ Test_LXTabCtrl
LXToolTip	提示框	Samples/Test_Wnd/Test_Ctrls/ Test_LXToolTip
LXTreeCtrl	树控件	Samples/Test_Wnd/Test_LXTreeCtrl
LXDialog	对话框	Samples/Test_Wnd/Test_LXDialog
LXColorDialog	颜色对话框	Samples/Test_Wnd/Test_CommonDialog/Test_LXColorDialog
LXFileDialog	文件对话框	Samples/Test_Wnd/Test_CommonDialog/Test_LXFileDialog
LXFontDialog	字体对话框	Samples/Test_Wnd/Test_CommonDialog/Test_LXFontDialog
	信息提示框	Samples/Test_Wnd/Test_CommonDialog/Test_LXMsgBox

窗口的创建

一般窗口或控件的创建

所有的窗口控件均可以通过窗口对象上的 CreateWindow()函数来创建。

```
CreateWindow(strWndName, strWndStyle, x, y, width, height, parent(可选), id(可选))
CreateWindow(strWndName, strWndStyle, rect, parent(可选), id(可选))
```

参数

参数名	类型	描述	是否可选	默认值
strWndName	string	窗口标题名称	否	无
strWndStyle	string	窗口风格	否	无
x	number	窗口左上角 x 值	否	无
x	number	窗口左上角 y 值	否	无
width	number	窗口宽度	否	无
height	number	窗口高度	否	无
rect	LXRect	窗口矩形大小	否	无
parent	窗口对象	窗口的父窗口	否	无
id	number	窗口 ID	是	0

返回值

类型	描述
窗口对象	创建成功返回一个有效的窗口对象 ,否则返回 nil

注意事项

strWndStyle 参数取值：
系统定义一个字符代表一个风格，一种方法是写入由一系列字符组合而成的字符串，比如’c’代表窗口是孩子，‘v’代表窗口可见，那么 strWndType 参数就可以是”vc”, 表示窗口是孩子而且可见。其余字符意义在下表中一一列出。
另一种方法是直接使用系统定义的宏，系统定义的宏含有特定意义，如下表。需要注意的是，系统定义的宏只是一些常用风格，若有特殊要求，必须使用前种方法。
1. 字符代表意义：

控件类型	strWndStyle
LXWnd	'C': 孩子窗口，不能和'p'或'P'公用
	'O': 是否包含标题栏，弹出窗口有效
	'P': 弹出窗口，不能和'c'或'C'公用
	'V': 窗口是否可见
	'L': 是否裁减孩子窗口占有区域
	'B': 是否包含边框
	'S': 是否有系统菜单
	'X': 是否裁剪兄弟窗口
	'T': 窗口是否透明
	'D': 窗口是否禁止
	'F': 窗口是否不接受焦点;
	'G': 窗口是否接收 Tab 键
	'N': 窗口是否无需激活
	'Z': 窗口是否可改变大小，弹出窗口有效
	'M': 窗口是否拥有系统菜单，弹出窗口有效

	'E': 是否拥有垂直滚动杠
	'H': 是否拥有水平滚动杠
	'Q': 改变窗口大小时是否绘制窗口内容，弹出窗口有效
	'A': 是否拥有极大/恢复按钮，弹出窗口有效
	'I': 是否拥有极小化按钮，弹出窗口有效
LXButton	'J': PUSH 按钮
	'R': RAIDO 按钮
	'K': CHECKBOX 按钮
LXComboBox	'J': 简单风格;
	'K': 下拉风格，可编辑
	'Y': 下拉列表风格，不能编辑
LXEdit	'W': 密码风格
	'R': 只读
LXGroupBox	无特殊风格
LXHeadCtrl	无特殊风格
LXListBox	'J': 多选风格
	'K': 排序
LXListCtrl	'R': 报表风格
	'K': 图标风格
	'Y': 整行选中
LXMenu	'Y': 含竖线
	'W': 不含竖线，但文字与左边界有一定距离
LXProgress	'W': 水平风格
	'R': 垂直风格
LXStatic	'W': 文字
	'K': 图像
	'U': 数字
LXSliderCtrl	与 LXProgress 相同
LXTabCtrl	无特殊风格
LXTooltip	无特殊风格
LXTreeCtrl	'K': 含 CheckBox
	'W': 含按钮
	'Y': 含竖线
	'U': 不能拖拽

2 系统定义的宏的意义：

控件类型	strWndStyle
LXWnd	**XWS_POPUP**,包含属性：弹出窗口，可见，含标题栏，边框，有“极大/恢复按钮”，有“拥有极小化按钮”。
	XWS_CHILD 包含属性：孩子窗口，可见，含边框。
LXButton	**XWS_PUSHBUTTON**,包含属性： 按钮，孩子窗口，可见，含边框。
	XWS_CHECKBOX_BUTTON,包含属性： CHECKBOX 按钮，孩子窗口，可见，含边框。

	XWS_RADIO,包含属性： Radio 按钮，孩子窗口，可见，含边框。
LXComboBox	**XWS_SIMPLE**,包含属性： 简单风格，孩子窗口，可见，含边框。
	XWS_DROPDOWN,包含属性： 下拉风格，可编辑，孩子窗口，可见，含边框。
	XWS_ DROPLIST,包含属性： 下拉列表风格，不能编辑，孩子窗口，可见，含边框。
LXEdit	**XWS_PASSWORD**,包含属性： 密码风格，孩子窗口，可见，含边框。
	XWS_READONLY,包含属性： 只读风格，孩子窗口，可见，含边框。
LXGroupBox	**XWS_CHILD,**包含属性：孩子窗口，可见，含边框。
LXHeadCtrl	**XWS_CHILD,**包含属性：孩子窗口，可见，含边框。
LXListBox	**XWS_MULTISEL**,包含属性： 多选风格，孩子窗口，可见，含边框。
	XWS_LISTSORT,包含属性： 排序，孩子窗口，可见，含边框。
LXListCtrl	**XWS_REPORT**,包含属性： 报表风格，孩子窗口，可见，含边框。
	XWS_ICON,包含属性： 图标风格，孩子窗口，可见，含边框。
	XWS_ROWSELECT,包含属性： 整行选中，孩子窗口，可见，含边框。
LXMenu	**XWS_VERTLINE**,包含属性： 含竖线，弹出窗口，可见，含边框。
	XWS_LEFTSPACE,包含属性： 不含竖线，但文字与左边界有一定距离 ，弹出窗口，可见，含边框。
LXProgress	**XWS_VERT**,包含属性：垂直风格，孩子窗口，可见，含边框。
	XWS_HORZ,包含属性：水平风格，孩子窗口，可见，含边框。
LXStatic	**XWS_TEXT**,包含属性：文字风格，孩子窗口，可见，含边框。
	XWS_IMAGE,包含属性：图像风格，孩子窗口，可见，含边框。
	XWS_NUMBER,包含属性：数字风格，孩子窗口，可见，含边框。
LXSliderCtrl	**XWS_VERT**,包含属性:垂直风格，孩子窗口，可见，含边框。
	XWS_HORZ,包含属性:水平风格，孩子窗口，可见，含边框。
LXTabCtrl	**XWS_CHILD,**包含属性：孩子窗口，可见，含边框。
LXTooltip	**XWS_TOOLTIP,**包含属性：弹出窗口，可见，含边框。
LXTreeCtrl	**XWS_CHECKBOX_TREE**,包含属性： 含 CheckBox，孩子窗口，可见，含边框。
	XWS_BUTTON_TREE,包含属性： 含按钮，孩子窗口，可见，含边框。
	XWS_LINE,包含属性： 含竖线，孩子窗口，可见，含边框。

XWS_NODROP,包含属性：不能拖拽，孩子窗口，可见，含边框。

示例

```
dlg = LXDialog.new(“..\\界面\\dlg.xrc”)    --新建对话框对象
dlg.Create()     --创建对话框
dlg.ShowWindow(true)    --显示对话框
button = LXButton.new() --新建按钮对象
button.CreateWindow ("TestButton", "cv", 0,0,100,20, dlg, 100) --创建按钮窗口
combobox = LXComboBox.new() --新建组合框对象
combobox.CreateWindow("TestComboBox",XWS_SIMPLE, 0,30,100,50,dlg, 101)
--创建组合框窗口
```

参见

```
Samples \Test_Wnd\Test_LXWnd\脚本\Main.lua
```

注意事项

创建窗口使用字符风格的时候, LXWnd 的风格是所有其它控件的共用风格, 所有其余的控件可以直接使用 LXWnd 的风格.其它控件列出的风格是控件自己特有的风格.

通过文件创建 **Dialog**

模态对话框

```
Dlg = LXDialog.new(strPath)      --新建对话框对象
local nResult = Dlg.DoModal()    --弹出模态对话框
```

参数

参数名	类型	描述	是否可选	默认值
strPath	string	Xrc 文件全路径或者相对于 lua 文件本身的相对路径	否	无

返回值

类型	描述
LXDialog	成功构造返回一个对话框对象，否则返回 nil

注意事项

示例

```
Dlg = LXDialog.new("..\\界面\\HelloXrc.xrc")
local nResult = Dlg.DoModal()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

非模态对话框

```
Dlg = LXDialog.new(strPath)   --新建对话框对象
Dlg.Create(Parent)     --创建对话框窗口
Dlg.ShowWindow(true) --显示对框窗口
```

参数

参数名	类型	描述	是否可选	默认值
strPath	string	Xrc 文件全路径或者相对于 lua 文件本身的相对路径	否	无
Parent	LXWnd	对话框的父窗口	是	nil

返回值

类型	描述
LXDialog	成功构造返回一个对话框对象，否则返回 nil

注意事项

示例

```
Dlg = LXDialog.new("..\\界面\\HelloXrc.xrc")
Dlg.Create()
Dlg.ShowWindow(true)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

Common Dialog

常用到的窗口如颜色对话框，字体对话框，文件对话框， 消息对话框

LXColorDialog 颜色对话框

参考 LXColorDialog

LXFileDialog 文件对话框

参考 LXFileDialog

LXFontDialog 字体对话框

参考 LXFontDialog

消息对话框

参考消息对话框

窗口的销毁

一般来说，窗口可以由 Lua 垃圾回收机制进行销毁。但为了提高效率，用户可以通过窗口对象 DestroyWindow()函数来销毁已经创建的窗口。

```
DestroyWindow()
```

参数

无

返回值

无

注意事项

示例

```
Dlg.DestroyWindow()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

窗口属性和行为

所有的窗口从 LXWnd 派生，因此 LXWnd 类的各个函数均可以被各个窗口/控件使用。详见 LXWnd 文档。用户可以通过 Lua 函数控制窗口的属性和行为。
以 LXEdit 为例，既可以调用 LXWnd 上的 MoveWindow()函数来移动 LXEdit 控件，也可以调用 LXEdit 的 Clear()函数来清空文本框里的内容。

窗口消息与交互

每一个对话框可以关联一个主 Lua 脚本：选中对话框，然后在对话框的属性窗口中选中主 Lua 脚本属性（可参考窗口示例）。
给窗口添加 Lua 脚本后，就可以给对话框的各控件添加消息响应函数：选中需要添加消息响应的控件,在相应的控件属性窗口中添加需要的控件消息，在对应的函数中用户自行添加消息处理代码。

1 窗口内的交互

直接调用窗口的 FireEvent()函数,或者直接调用窗口函数. 比如:

```
#include "XMain.h"
loadlib("XLuaLib")
loadlib("XXRCCreator")

ON_XEVENT(20000, "20000", "OnTrigger1")
 --{{XFUN EVENT
      --在此处添加事件
      X_ON_BUTTONCLICK(OnButtonClick1,1)
```

```
--}}XFUN EVENT



-------------------------------------------
--事件处理函数
-------------------------------------------
function   OnButtonClick1(this,nEventID,SendName,param)
      --在此处添加事件相应代码
      this.FireEvent(20000, "20000", param)
      OnTrigger1(this,nEventID,SendName,param)
end

function   OnTrigger1(this, nID, sendname, param)
      print("OnTrigger1 Called")
end
```

2 窗口间的交互

2.1 使用窗口的返回值.

dlg.lua

```
X_ON_BUTTONCLICK(OnButtonClick3,3)
function OnButtonClick3(this,nEventID,SendName,param)
      --在此处添加事件相应代码
      dlg1 = LXDialog.new("..\\界面\\dlg1.xrc")
      local nResult = dlg1.DoModal()
      if nResult == IDOK then
            OnTrigger1(this, nEventID, SendName, param)
      end
end
```

dlg1.lua

```
#include "XMain.h"
loadlib("XLuaLib")
loadlib("XXRCCreator")

--{{XFUN EVENT
      --在此处添加事件
      X_ON_BUTTONCLICK(OnButtonClick1,1)
--}}XFUN EVENT



-------------------------------------------
--事件处理函数
```

```
-------------------------------------------
function OnButtonClick1(this,nEventID,SendName,param)
	--在此处添加事件相应代码
	this.EndDialog(IDOK)
end
```

2.2: 使用窗口的 FireEvent()函数

dlg.lua

```
#include "XMain.h"
loadlib("XLuaLib")
loadlib("XXRCCreator")

ON_XEVENT(20000, "20000", "OnTrigger1")
X_ON_BUTTONCLICK(OnButtonClick4,4)
function OnButtonClick4(this,nEventID,SendName,param)
	--在此处添加事件相应代码
	dlg2 = LXDialog.new("..\\界面\\dlg2.xrc")
	dlg2.dlg = this
	dlg2.DoModal()
end

function   OnTrigger1(this, nID, sendname, param)
	print("OnTrigger1 Called")
end
```

dlg2.lua

```
#include "XMain.h"
loadlib("XLuaLib")
loadlib("XXRCCreator")

--{{XFUN EVENT
	--在此处添加事件
	X_ON_BUTTONCLICK(OnButtonClick1,1)
--}}XFUN EVENT



-------------------------------------------
--事件处理函数
-------------------------------------------
function OnButtonClick1(this,nEventID,SendName,param)
	--在此处添加事件相应代码
	dlg = this.dlg
	dlg.FireEvent(20000, "20000", param)
```

```
    this.EndDialog(0)
end
```

3 窗口和与主对象间的交互

XMain.lua

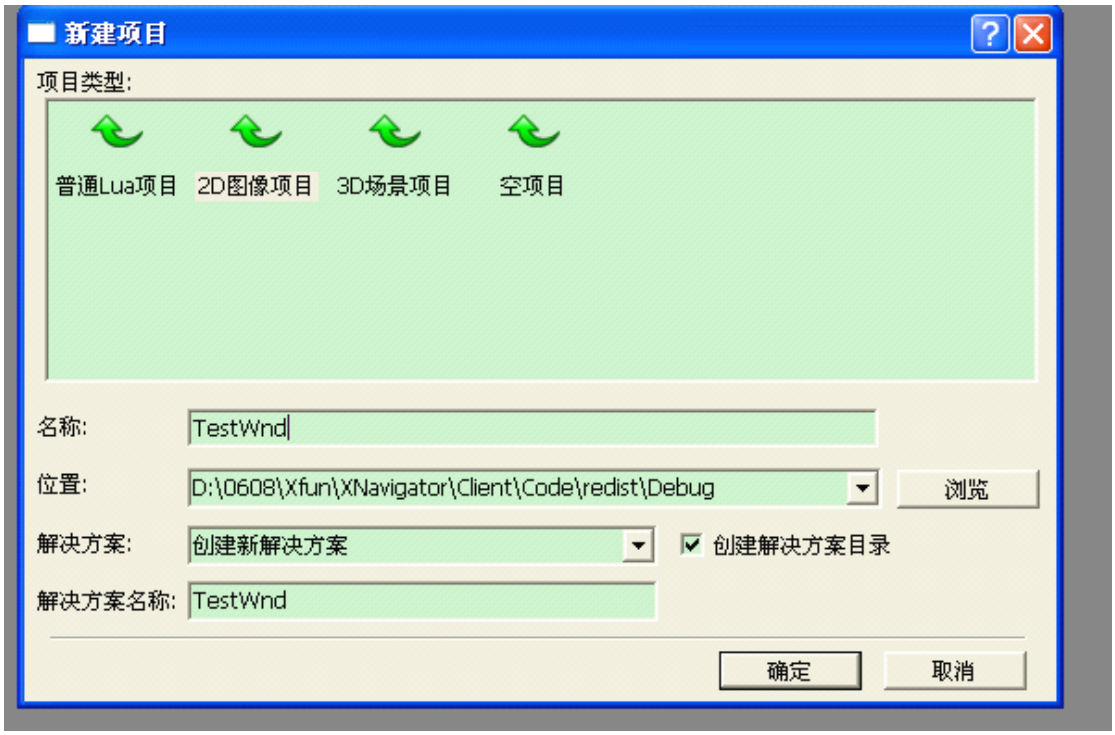
```
ON_XEVENT(20001, "20001", "OnMainTrigger")

function   OnMainLoadLua(this,nEventID,SendName,param)
    dlg = LXDialog.new("..\\界面\\dlg.xrc")
    dlg.MainObject = this
    dlg.DoModal()
end

function   OnMainTrigger(this, nID, sendname, param)
    print("OnMainTrigger Called")
end
```

dlg.lua

```
function OnButtonClick2(this,nEventID,SendName,param)
    --在此处添加事件相应代码
    mainobject = this.MainObject
    mainobject.FireEvent(20001, "20001", param)
end
```

窗口示例

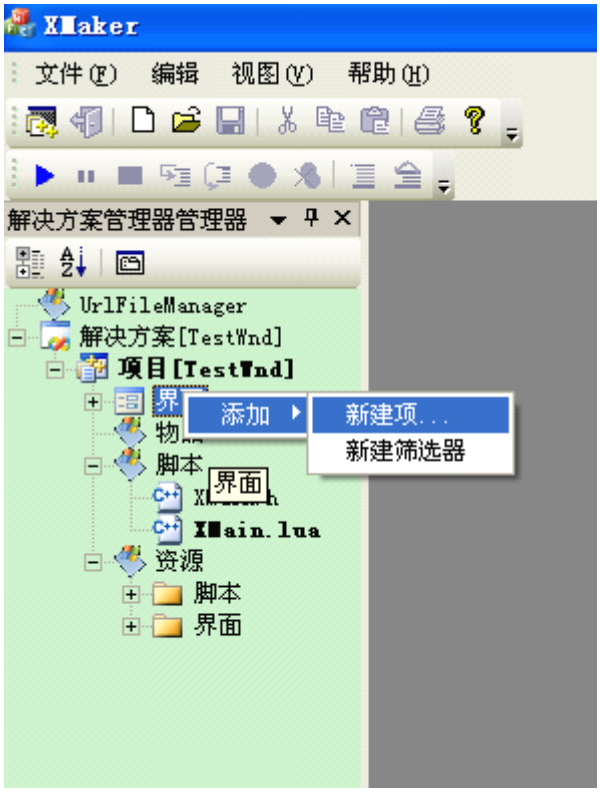
1 运行 XMake.exe
依次选中菜单文件→新建→项目→2D 图像项目, 然后输入项目名称, 取名 TestWnd, 界面如下:

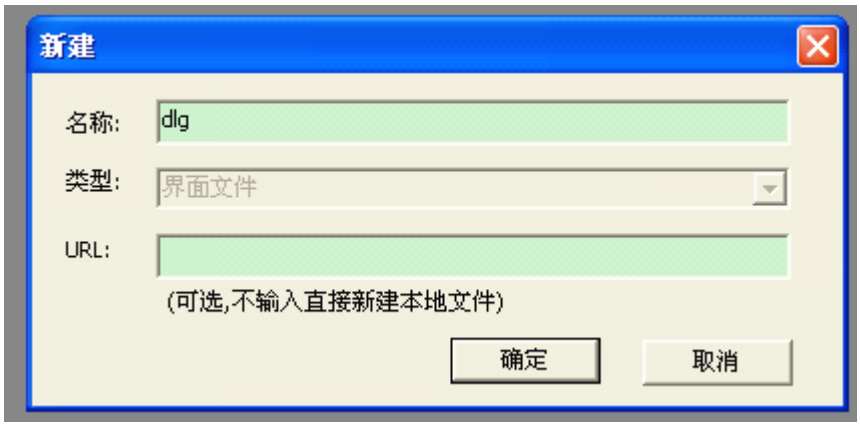


然后点击”确定”按钮.此时已经系统默认生成了 XMain.h 和 XMain.lua 两个文件,. XMain.h 包含了系统定义的消息宏.XMain.lua 包含了主对象处理的消息处理函数，你只需要在每个函数里处理你需要的代码就可。

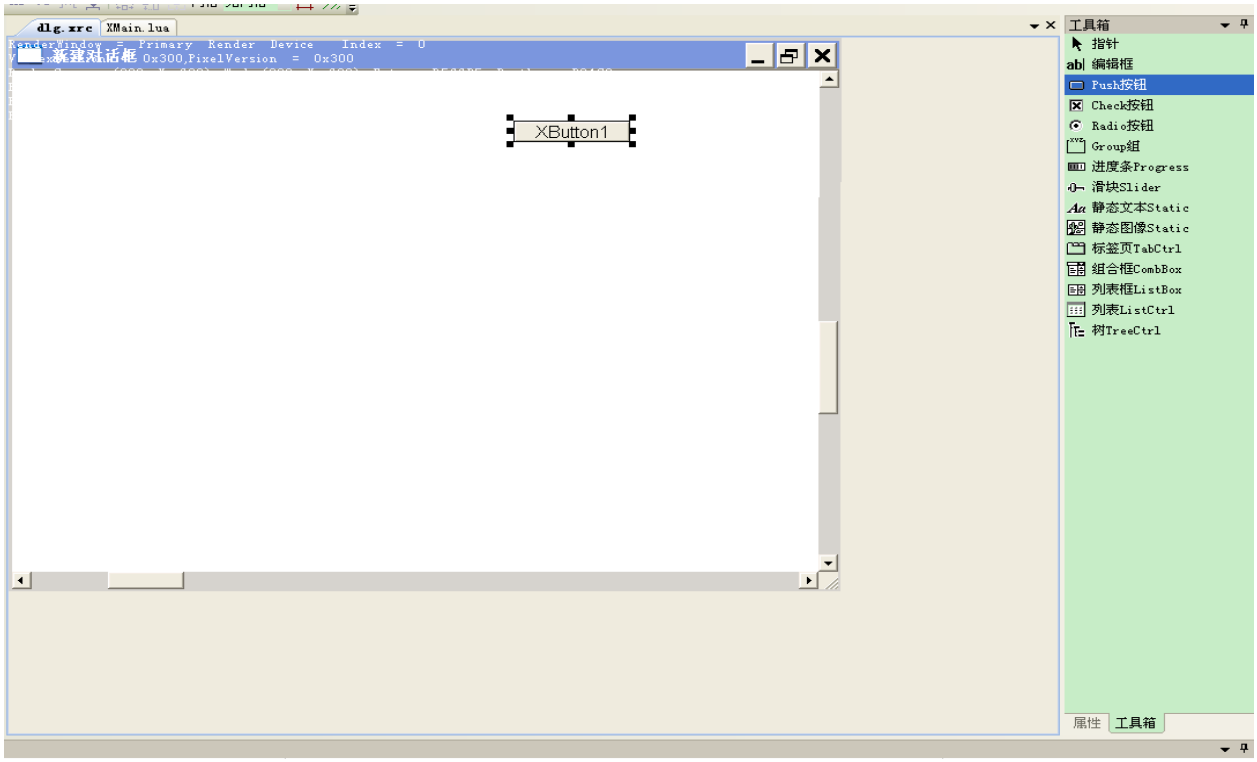
2 添加对话框资源

选中解决方案管理器面板，打开项目[TestWnd]，选中”界面”，然后右键→添加→新建项，然后输入对话框资源名称 dlg，界面如下:

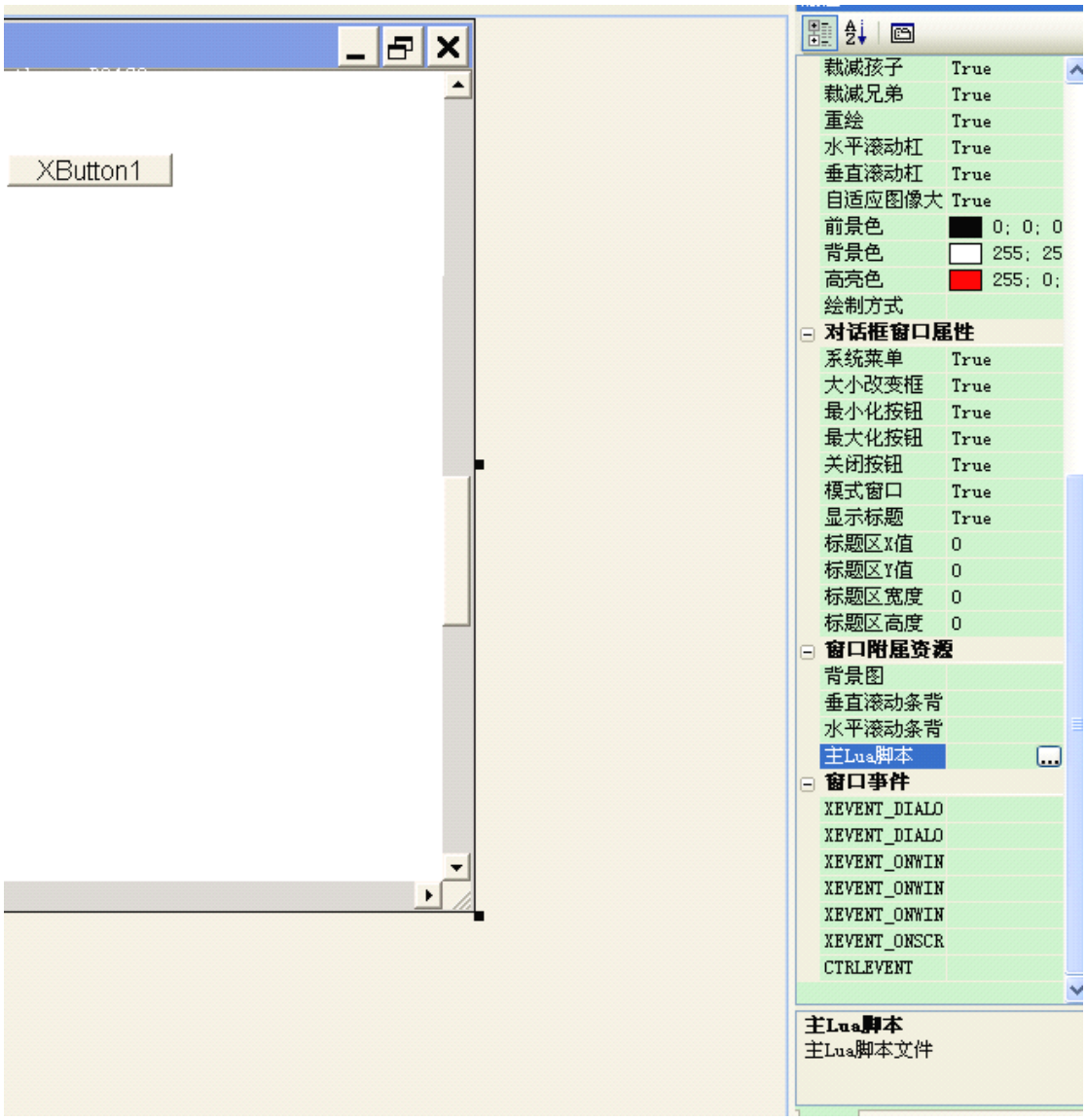




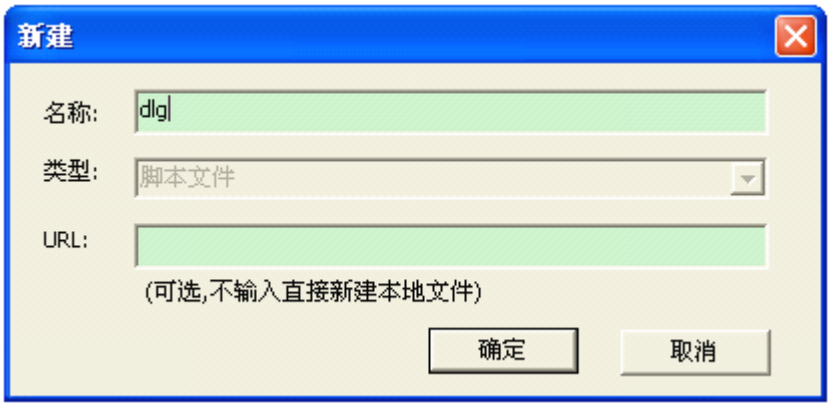
此时系统会生成一个空的对话框资源，用户可以给对话框添加控件。操作如下:
选中工具箱，将需要的控件选中拖入左边对话框。



3 给对话框添加脚本
选中对话框 dlg.xrc，然后选中属性面板，点击主 Lua 脚本项



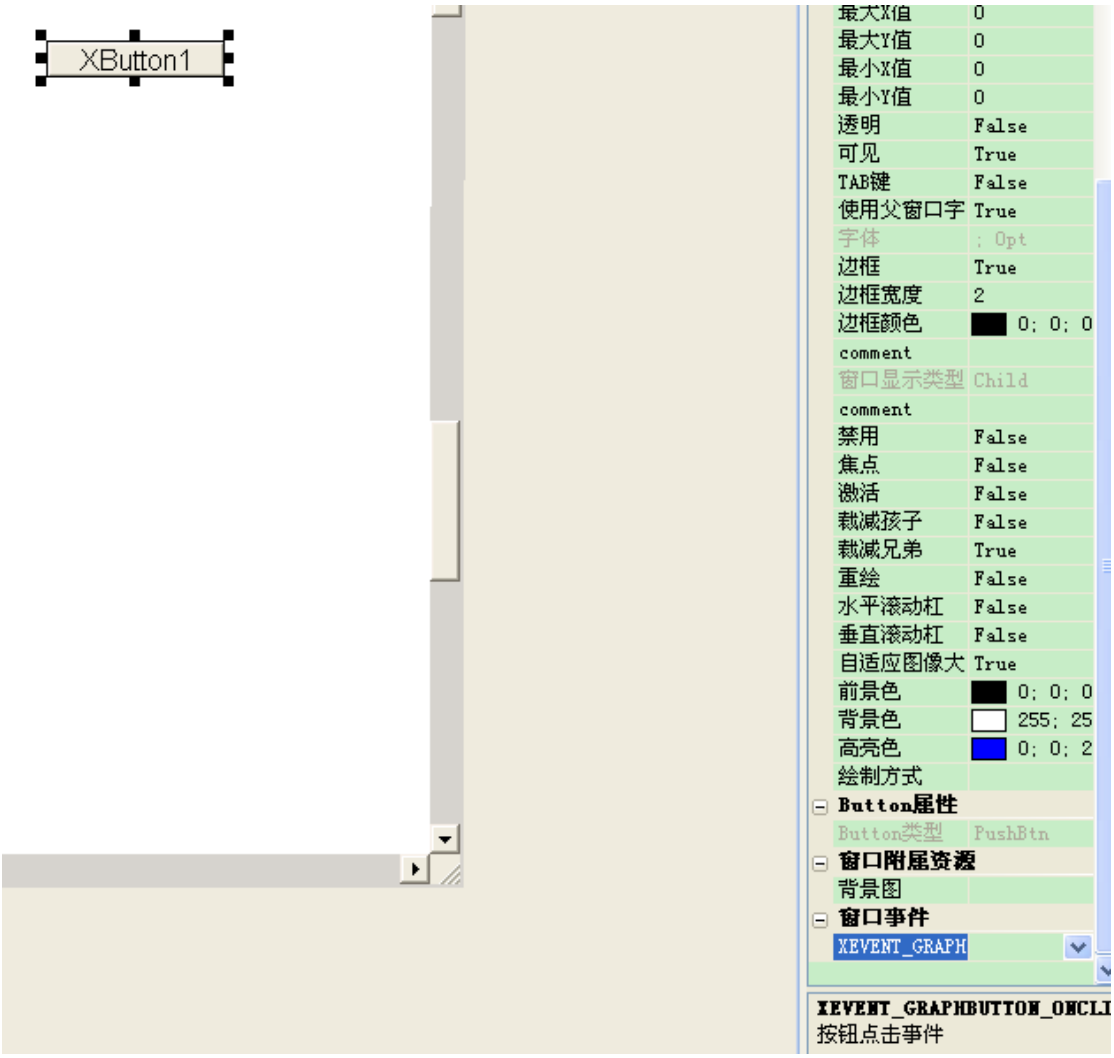
然后单击右边的按钮，输入脚本名称



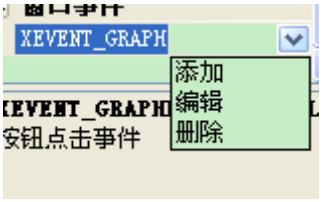
添加完后，解决方案面板中脚本项下会添加一个 dlg.lua 文件，双击可以打开此文件。

4 给控件添加消息处理函数

选中需要添加消息的控件，如选中 XButton1



单击右边的按钮，出现菜单如下:



添加: 给控件添加一个消息处理函数
编辑: 直接跳转到消息处理函数里面去
删除: 将消息处理函数注释掉

选择添加后 dlg.lua 会生成消息函数处理函数，如下图所示

```
#include "XMain.h"


--------------------------------------------
--事件处理函数
--------------------------------------------
X_ON_BUTTONCLICK(OnButtonClick1,1)
function OnButtonClick1(this,nEventID,SendName,param)
    --在此处添加事件相应代码
end
```

之后用户就可以在消息处理函数里面添加事件处理代码。

2.4.4 脚本编辑

脚本编辑指对 Lua 脚本的编辑，运行与调试。

脚本编辑是整个应用的核心，应用的逻辑与部分数据都是通过脚本编辑来产生。

Lua 代码编写

XMaker 开发平台提供了 Lua 语言的编译，调试，与运行环境。

XMaker 提供了一套 API。它扩展了标准 Lua 语言的 API 函数，比如提供了图像的绘制，声音的播放，窗口或场景的创建等。

下面是 XMaker 中使用 Lua 的一些常见方法。
1. 宏定义的使用。
XMaker 支持使用宏，比如 XMain.h 里定义了
#define HELLOWORLD “Hello, world!”
则可使用 print(HELLOWORLD)代替 print(“Hello, world!”)

2. 使用#include ”xxx.h”包含自定义头文件
比如#include "XMain.h"。

3. 使用 loadlib(“libname”)装载库文件
比如 loadlib("XLuaLib")。一般来说系统已经默认加载了相关库文件，用户也可以根据需要自行加载库文件。

4. 使用 requrie “xxx.lua” 装载其它 lua 文件
比如 require “other.lua”。

5. 其它代码的编写，参考 API。

Lua 代码调试运行

编辑完脚本后，用户可以通过启动调试来调试应用。参见调试。

2.4.5 物品编辑

编辑物体的基本信息。针对 3D 场景项目。

2.4.6 图像编辑

实现 X 魔方支持的图像格式文件的编辑

2.4.7 模型编辑

实现 X 魔方支持的模型格式文件的编辑

2.4.8 骨架编辑

实现 X 魔方支持的骨架格式文件的编辑

2.4.9 动画编辑

实现 X 魔方支持的动画格式文件的编辑

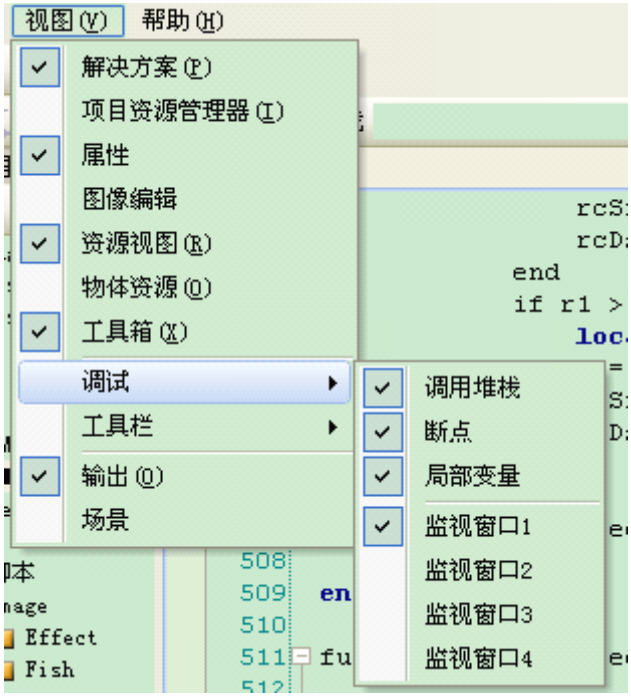
2.4.10 材质编辑

实现 X 魔方支持的材质格式文件的编辑

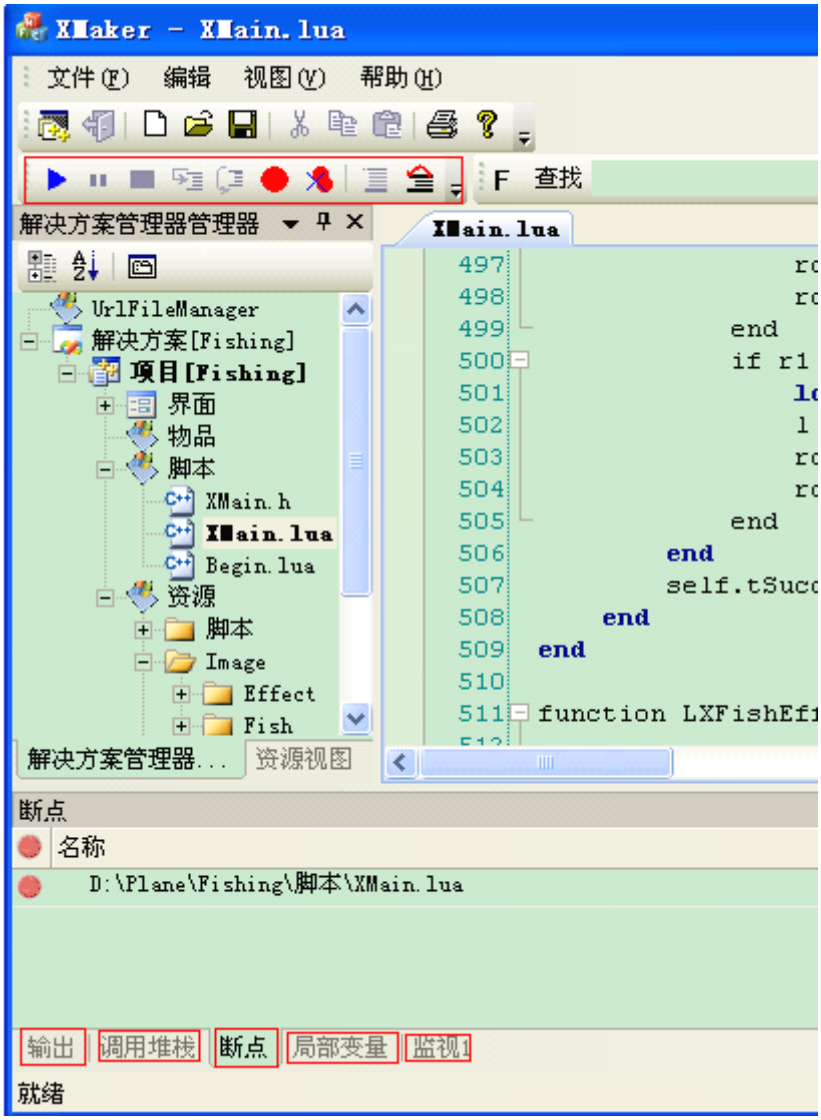
2.5 调试

XMaker 不仅提供了 Lua 的编译和运行环境，而且提供了强大的调试环境。

调试环境包括调试工具栏,输出窗口,调用堆栈窗口，断点窗口，局部变量窗口以及四个变量监视窗口。这些视图的显示或关闭都可以在在菜单的视图菜单中控制。如下图所示：



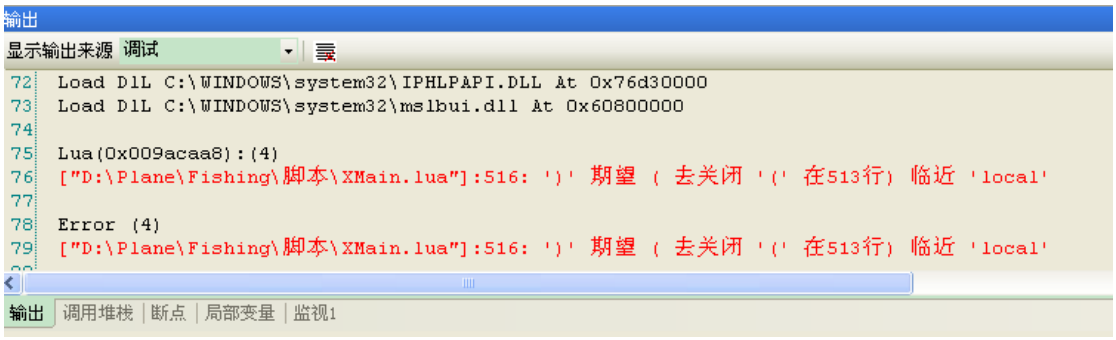
调试工具栏



如上图所示，它包含以下几个功能:
开始调试(F5) 运行 lua 代码,调试过程中的输出信息将在输出窗口中打印
全部中断 程序将在程序当前运行代码处中断
停止调试(Ctrl+Break)
逐语句调试(F10)
逐过程调试(F11)
切换断点(F9)
删除所有断点(Ctrl+Shift+F9)

输出窗口

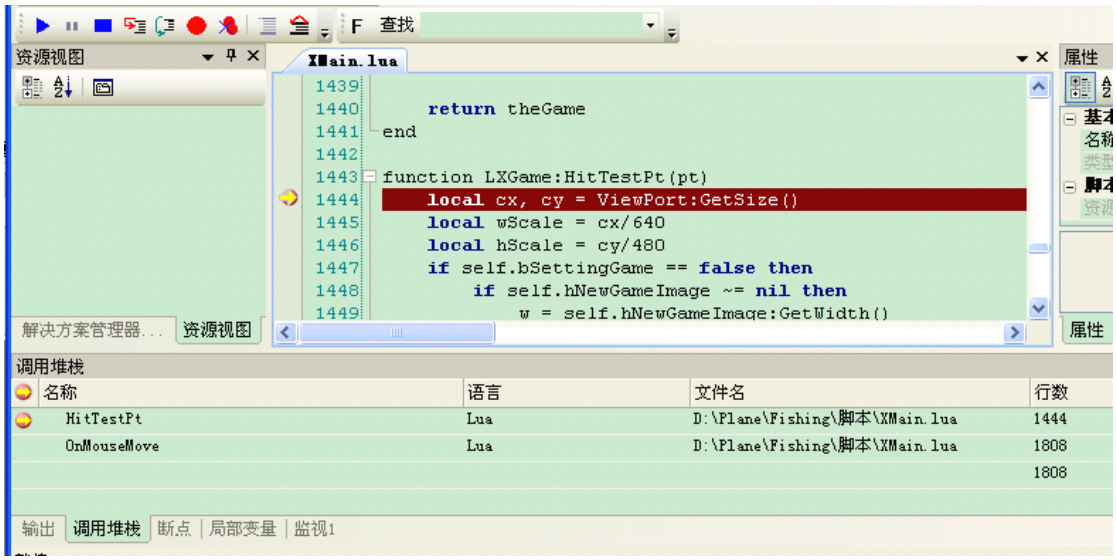
如下图所示，程序的调试中的信息将出现在输出窗口中,包含程序启动装载的库,程序运行的错误信息等



程序出错的信息将以红色字体出现,双击该行将跳到程序出错的代码行上

调用堆栈

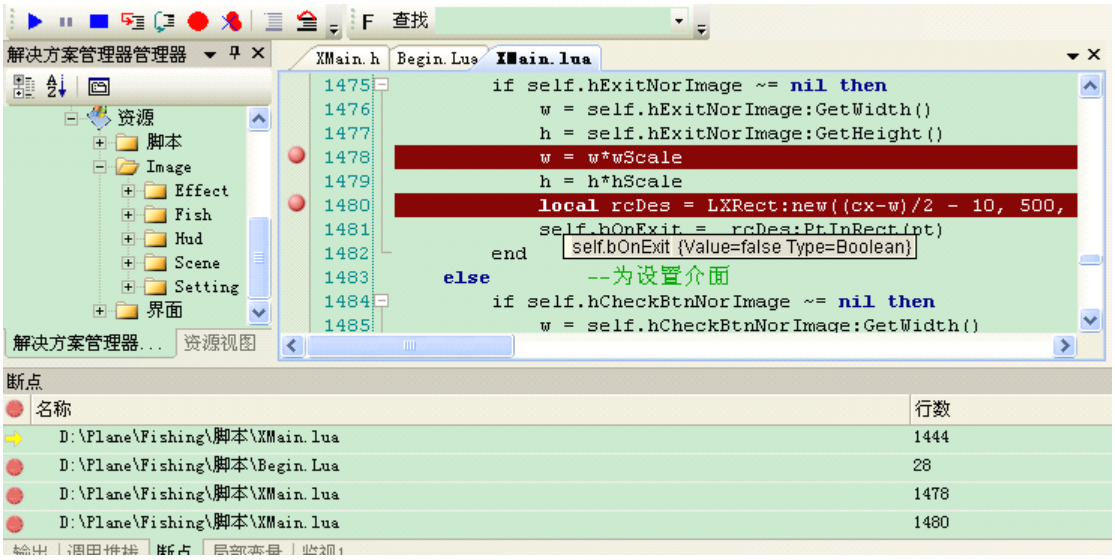
调用堆栈将当前的调用层次关系列出,如图:



双击指定的层将跳到它的层次代码函数上。

断点窗口

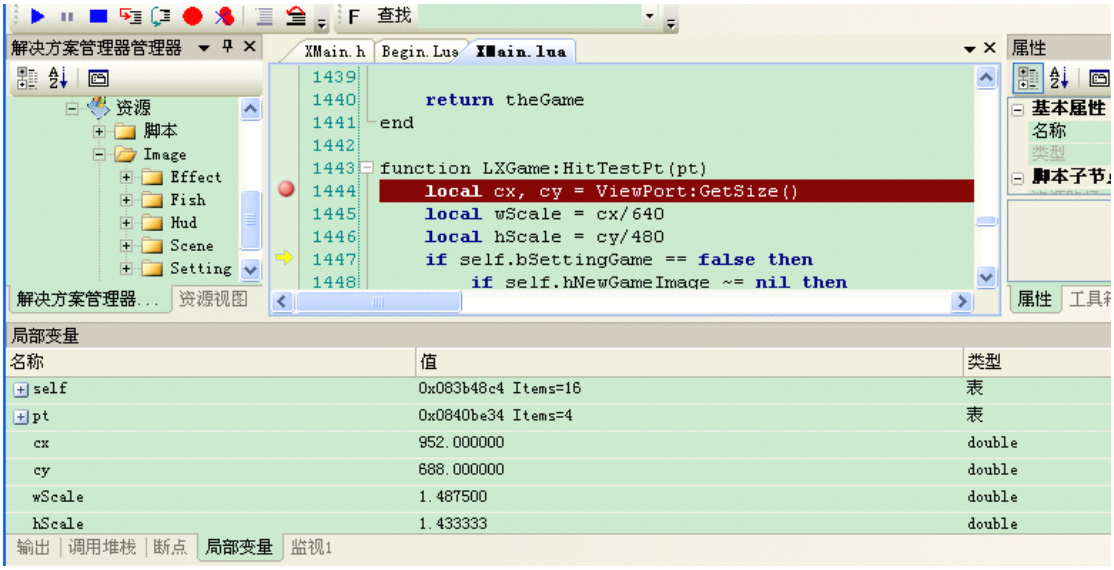
当前项目的所有的断点信息将在该窗口中列出



双击指定的的列将跳到其断点的代码上

局部变量窗口

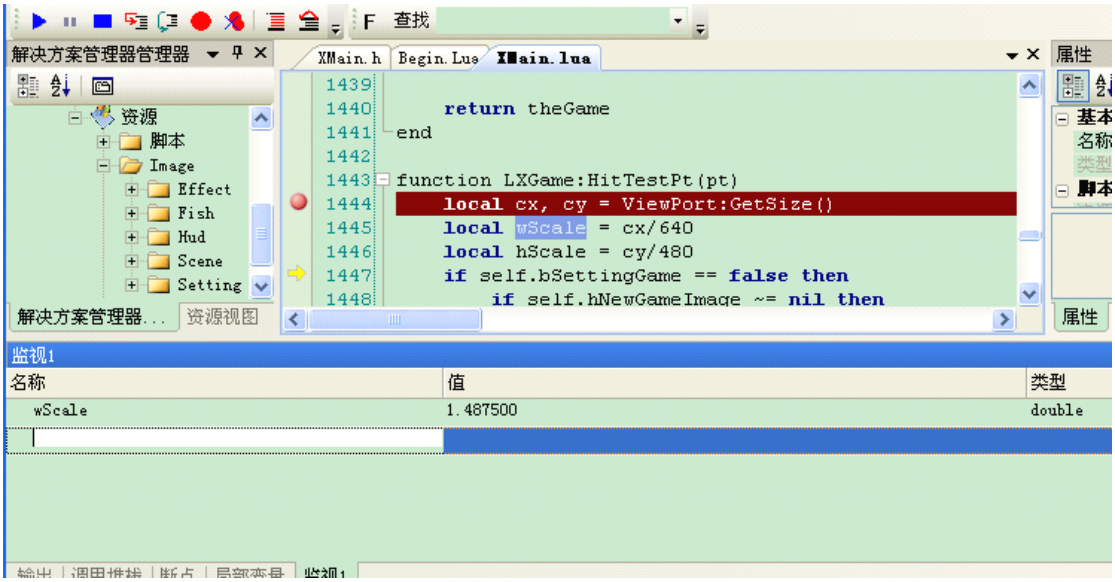
将当前调试点的局部变量信息列出



可以查看当前运行点的局部变量的值

监视窗口

监视窗口可以查看指定的变量的当前值,可将变量拖到监视窗口中，或在监视窗口的列名称中双击写入变量名称.变量的值和类型将列出 如下图:



3 开发 API

3.1 概述

本部分内容主要介绍 X 魔方编辑器提供给用户在编写脚本程序过程中使用的各种常用 Lua 导出类、Lua 导出函数及事件，这些导出类和导出函数是对 Lua 通用 API 的扩展，使 Lua 的使用更加方便。通过这些类、函数、事件用户可以向 X 魔方加入各种不同的元素，如图像、声音、窗口、物品等，并通过不同的事件响应建立起各个元素之间的逻辑联系，实现相应的功能。

3.2 宏

Lua 语言本身并不支持宏定义，做为 Lua 语言的扩展，X 魔方编辑器为用户提供了宏定义的功能。使用宏可提高程序的通用性和易读性，减少不一致性，减少输入错误和便于修改。

如何定义宏

宏定义的格式如下:

```
#define 宏名(参数表) 表达式
```

#define、宏名、表达式构成了一个宏定义，
用户可以在 X 魔方编辑器中定义自己的宏，当用户新建一个项目之后，解决方案面板中的脚本节点会包含一个名为 XMain.h 的头文件，这个头文件包含了 X 模仿编辑器已经定义的系统宏，用户只需在这个头文件中按照格式定义自己的宏就可以在这个项目的所有 Lua 脚本中使用。也可以在脚本中自行添加宏。

3.3 内置对象与内置函数

内置对象与内置函数的特点在于对象或函数已经内置，可以直接使用。

内置对象

__MainObject

功能描述

该对象实现 Lua 主对象封装类。

函数列表

函数名称	说明
GetViewPortSize	获得视图窗口的大小
SetViewPortSize	设置视图窗口的大小
ExitApp	退出主程序

GetViewPortSize

获得视图窗口的大小

```
GetViewPortSize()
```

参数

```
无
```

返回值

参数类型	说明
number	视图窗口的宽度
number	视图窗口的高度

注意事项

```
无
```

示例

```
Width, Height = __MainObject.GetViewPortSize()
print(Width)
print(Height)
```

参见

无

SetViewPortSize

设置视图窗口的大小

```
SetViewPortSize(nWidth, nHeight, nDepth, bFullScreen)
```

参数

参数名	类型	说明	是否可选	默认值
nWidth	number	视图窗口宽度	否	无
nHeight	number	视图窗口高度	否	无
nDepth	number	颜色深度	是	16
bFullScreen	boolean	是否全屏	是	false

返回值

参数类型	说明
boolean	设置是否成功

注意事项

颜色深度取值如下:
= 16　　16 位色
= 32　　32 位色

示例

```
bFullScreen = false
--鼠标单击切换全屏，窗口状态
function OnLButtonDown(this, EventID, SendName, param)
	if not bFullScreen then
		this.SetViewPortSize(1024, 768, 16, true)
		bFullScreen = true
	else
		this.SetViewPortSize(1024, 768)
		bFullScreen = false
	end
```

```
end
```

参见

```
无
```

ExitApp

退出主程序

```
ExitApp
```

参数

```
无
```

返回值

参数类型	说明
boolean	退出是否成功

注意事项

```
无
```

示例

```
function OnKeyDown(this, EventID, SendName, param)
    --如果按下回车键，则退出主程序
    if param == 0x0D then
        this.ExitApp()
    end
end
```

参见

```
无
```

内置函数

功能描述

实现基本的系统功能

函数列表

函数名称	说明
require	包含一个 Lua 文件
loadlib	加载 Lua 库
FireEvent	触发对象的事件
print	打印变量信息

require

包含一个 Lua 文件，别名 include

```
require(FileName)
```

参数

参数名	类型	说明	是否可选	默认值
FileName	string	指定的 Lua 文件	否	无

返回值

无

注意事项

无

示例

```
require(“game.lua”)
```

参见

无

loadlib

加载一个 C 动态库形式的 Lua 库

```
loadlib(FileName)
```

参数

参数名	类型	说明	是否可选	默认值
FileName	string	指定的 Lua 库名	否	无

返回值

无

注意事项

无

示例

```
loadlib(“XLuaLib”)
loadlib(“XLuaBase”)
```

参见

无

FireEvent

触发一个当前脚本的事件

```
FireEvent(EventID, SendName, Param)
```

参数

参数名	类型	说明	是否可选	默认值
EventID	number	事件 ID	否	无
SendName	string	事件发生的对象	否	无
Param	任意类型	事件参数	否	无

返回值

无

注意事项

无

示例

```
FireEvent(1, “*”, nil)
```

参见

无

print

在调式环境中打印变量值

```
print(Var)
```

参数

参数名	类型	说明	是否可选	默认值
Var	任意类型	需要打印的变量	否	无

返回值

无

注意事项

无

示例

```
print(“Hello World”)
print(100)
```

参见

无

3.4 常用对象

3.4.1 基本对象

本部分介绍了较为常用的 Lua 导出类对象。

LXPoint

功能描述

该对象实现对点的各种操作。

属性列表

属性名称	说明
x	LXPoint 的 x 属性
y	LXPoint 的 y 属性

示例

```
pt = LXPoint.new(20, 50)
print(pt.x)
print(pt.y)
pt.x = 100
pt.y = 200
print(pt.x)
print(pt.y)
```

函数列表

函数名称	说明
new	创建一个 LXPoint 对象
GetValue	得到点的值
SetValue	设置点的值
SetX	设置点的 X 分量值
SetY	设置点的 Y 分量值
GetX	得到点的 X 分量值
GetY	得到点的 Y 分量值

new

创建一个 LXPoint 对象

```
new()
new(pt)
new(x , y)
```

参数

参数名	类型	描述	是否可选	默认值
pt	LXPoint	LXPoint 对象	否	无

参数名	类型	描述	是否可选	默认值
x	number	x 值	否	无
y	number	y 值	否	无

返回值

类型	描述
LXPoint	LXPoint 对象

注意事项

无

示例

```
Pt1 = LXPoint.new()                  --x = 0    y = 0
Pt2 = LXPoint.new(Pt1)               --x = 0    y = 0
Pt3 = LXPoint.new(50,60)             --x = 50    y = 60
```

参见

Samples\Test_LuaBase\Test_LXPoint\脚本\XMain.lua

GetValue

得到点的值

```
GetValue()
```

参数

无

返回值

类型	描述
number	返回 X 分量值
number	返回 Y 分量值

注意事项

无

示例

```
Pt = LXPoint.new(50,60)              --x = 50    y = 60
x , y = Pt.GetValue()
print(x)                              --输出 50
print(y)                              --输出 60
```

参见

Samples\Test_LuaBase\Test_LXPoint\脚本\XMain.lua

SetValue

设置点的值

```
SetValue(x , y)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	x 值	否	无
y	number	y 值	否	无

返回值

无

注意事项

无

示例

```
Pt = LXPoint.new()              --x = 0    y = 0
Pt.SetValue(50,60)
x , y = Pt.GetValue()
print(x)                                --输出 50
print(y)                                --输出 60
```

参见

Samples\Test_LuaBase\Test_LXPoint\脚本\XMain.lua

SetX

设置 x 值

```
SetX(xValue)
```

参数

参数名	类型	描述	是否可选	默认值
xValue	number	x 值	否	无

返回值

无

注意事项

无

示例

```
Pt = LXPoint.new()              --x = 0    y = 0
Pt.SetX(50)
x , y = Pt.GetValue()
print(x)                                --输出 50
print(y)                                --输出 0
```

参见

```
Samples\Test_LuaBase\Test_LXPoint\脚本\XMain.lua
```

SetY

设置 y 值

```
SetY(yValue)
```

参数

参数名	类型	描述	是否可选	默认值
yValue	number	y 值	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Pt = LXPoint.new()              --x = 0    y = 0
Pt:SetY(50)
x , y = Pt.GetValue()
print(x)                                --输出 0
print(y)                                --输出 50
```

参见

```
Samples\Test_LuaBase\Test_LXPoint\脚本\XMain.lua
```

GetX

得到 x 坐标

GetX()

参数

无

返回值

类型	描述
number	返回 X 分量值

注意事项

无

示例

```
Pt = LXPoint.new()                --x = 0    y = 0
x = Pt.GetX()
print(x)                                  --输出 0
```

参见

Samples\Test_LuaBase\Test_LXPoint\脚本\XMain.lua

GetY

得到 y 坐标

GetY()

参数

无

返回值

类型	描述
number	返回 y 分量值

注意事项

无

示例

```
Pt = LXPoint.new()                --x = 0    y = 0
y = Pt.GetY()
```

```
print(y)                                    --输出 0
```

参见

```
Samples\Test_LuaBase\Test_LXPoint\脚本\XMain.lua
```

LXRect

功能描述

该对象实现对矩形各种操作。

属性列表

属性名称	说明
left	矩形对象左上角 X 坐标值
top	矩形对象左上角 Y 坐标值
right	矩形对象右下角 X 坐标值
bottom	矩形对象右下角 Y 坐标值

示例

```
rc = LXRect.new(0, 0, 100, 100)
print(rc.left)              --输出 0
print(rc.top)               --输出 100
print(rc.right)             --输出 0
print(rc.bottom)            --输出 100
```

函数列表

函数名称	说明
new	创建一个 LXRect 对象
GetLeft	获得矩形左上角 left 值
GetTop	获得矩形左上角 top 值
GetRight	获得矩形右下角 right 值
GetBottom	获得矩形右下角 bottom 值
GetValue	返回矩形四左上角(left,top)右下角(right,bottom)四个值
TopLeft	返回矩形左上角 left,top 二个值

CenterValue	返回矩形中心点的横纵坐标值
Copy	复制指定的矩形值到当前矩形
Deflate	向中心点缩小指定的大小
Equal	判断与指定的矩形是否相等
Height	计算矩形的高度
Inflate	增加矩形的高和宽
Intersect	矩形为二个指定的矩形的交叉部份
IsEmpty	判断矩形是否为空
IsNull	判断矩形左上角和右下角的四个值是否都为 0
MoveToX	将矩形向左移动指定值的正值
MoveToXY	将矩形向左下方向移动指定值的正值
MoveToY	将矩形向下方向移动指定值的正值
Normalize	格化矩形使其高和宽都为正值
Offset	通过指定的值移动矩形
PtInRect	判断指定的点是否在矩形中
SetValue	设置矩形
SetEmpty	设置矩形为空矩形 (左上角和右下角的四个值都为 0)
Size	返回矩形的宽和高
Subtract	矩形为二指定矩形相减的值
BottomRight	返回矩形右下角 right,bottom 二个值
Union	矩形为指定二矩形的联合后的矩形
Width	计算矩形的宽度

new

创建一个 LXRect 对象

```
new ()
new(ptLeftTop, ptRightBottom)
new(pt, nWidth, nHigth)
new(left, top, right, bottom)
```

参数

参数名	类型	描述	是否可选	默认值
ptLeftTop	LXPoint	左上点坐标	否	无
ptRightBottom	LXPoint	右下点坐标	否	无

参数名	类型	描述	是否可选	默认值
pt	LXPoint	左上点坐标	否	无
nWidth	number	矩形宽度	否	无
nHeigth	number	矩形高度	否	无

参数名	类型	描述	是否可选	默认值
left	number	左坐标	否	无
top	number	上坐标	否	无
right	number	右坐标	否	无
bottom	number	下坐标	否	无

返回值

参数名	类型	描述
rect	LXRect	创建的矩形对象

注意事项

无

示例

```
--创建空矩形  坐标为(0,0,0,0)
Rect1 = LXRect.new()
--通过坐标创建矩形
Rect2 = LXRect.new(50, 50, 100, 100)
pt1 = LXPoint.new(50,20)
pt2 = LXPoint.new(80 ,90)
--通过左上点坐标和宽度高度创建矩形
Rect3 = LXRect.new(pt1, 100,100)
--通过左上点和右下点坐标创建矩形
Rect4 = LXRect.new(pt1,pt2)
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

GetLeft

获得矩形左上点 left 值

```
GetLeft()
```

参数

无

返回值

类型	描述

number	矩形左上点 X 方向坐标值

注意事项

无

示例

```
Rect = LXRect.new(10, 50, 100, 200)
left = Rect.GetLeft()
print(left)          --输出 10
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

GetTop

获得矩形左上角 top 值

```
GetTop()
```

参数

无

返回值

类型	描述
number	矩形左上点 Y 方向坐标值

注意事项

无

示例

```
Rect = LXRect.new(10, 50, 100, 200)
Top = Rect.GetTop()
print(Top)          --输出 50
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

GetRight

获得矩形右下角 right 值

```
GetRight ()
```

参数

无

返回值

类型	描述
number	矩形右下点 X 方向坐标值

注意事项

无

示例

```
Rect = LXRect.new(10, 50, 100, 200)
Top = Rect.GetTop()
print(Top)          --输出 100
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

GetBottom

获得矩形右下角 bottom 值

```
GetBottom ()
```

参数

无

返回值

类型	描述
number	矩形右下点 Y 方向坐标值

注意事项

无

示例

```
Rect = LXRect.new(10, 50, 100, 200)
bottom = Rect.GetBottom()
print(bottom)          --输出 200
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

GetValue

返回矩形四左上角(left,top)右下角(right,bottom)四个值

```
GetValue()
```

参数

无

返回值

类型	描述
number	矩形左上点 X 方向坐标值
number	矩形左上点 Y 方向坐标值
number	矩形右下点 X 方向坐标值
number	矩形右下点 Y 方向坐标值

注意事项

无

示例

```
Rect = LXRect.new(10, 50, 100, 200)
left,top,right,bottom = Rect.GetValue()
print(left)            --输出 10
print(top)              --输出 50
print(right)           --输出 100
print(bottom)         --输出 200
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

BottomRight

返回矩形右下角坐标点

```
BottomRight ()
```

参数

无

返回值

类型	描述
LXPoint	矩形右下点坐标值

注意事项

无

示例

```
Rect = LXRect.new(10, 50, 100, 200)
ptBottomRight = Rect.BottomRight()
print(ptBottomRight.GetX())      --输出 100
print(ptBottomRight.GetY())      --输出 200
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

CenterValue

返回矩形中心点的坐标

```
CenterValue ()
```

参数

无

返回值

类型	描述
LXPoint	矩形中心点坐标值

注意事项

无

示例

```
Rect = LXRect.new(10, 10, 100, 200)
ptCenter = Rect.CenterValue()
print(ptCenter.GetX())      --输出 55
print(ptCenter.GetY())      --输出 105
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Copy

复制指定的矩形值到当前矩形

```
Copy (rect)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	指定矩形	否	无

返回值

无

注意事项

无

示例

```
Rect = LXRect.new()
rectTemp = LXRect.new(100,100,200,200)
Rect.Copy(rectTemp)
--Rect 当前的四个值为 100,100,200,200
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Deflate

向中心点缩小指定的大小

```
Deflate (x, y)
Deflate(rect)
Deflate(left, top, right, bottom)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	左上点坐标向矩形中心减小的值	否	无
y	number	右下点坐标向矩形中心减小的值	否	无

参数名	类型	描述	是否可选	默认值
rect	LXRect	矩形以 rect 的四个值向矩形中心减小	否	无

参数名	类型	描述	是否可选	默认值
left	number	左上点 X 坐标减小值	否	无
top	number	左上点 Y 坐标减小值	否	无
right	number	右下点 X 坐标减小值	否	无
bottom	number	右下点 Y 坐标减小值	否	无

返回值

无

注意事项

无

示例

```
rect = LXRect.new(10, 10, 50, 50)
rect.Deflate(1, 2)    --rect 的四个值变为 11,12,49,48

rect1 = LXRect.new(10, 10, 50, 50)
rectDef = LXRect.new(1, 2, 3, 4)
rect1.Deflate(rectDef)    --rect1 的四个值变为 11,12,47,46
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Equal

判断与指定的矩形是否相等

```
Equal (rect)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	指定矩形	否	无

返回值

类型	描述
Boolean	两个矩形是否相等

注意事项

无

示例

```
rect = LXRect.new(10, 10, 50, 50)
rect.Deflate(1, 2)                    --rect 的四个值变为 11,12,49,48
rect1 = LXRect.new(10, 10, 50, 50)
bEqual = rect.Equal(rect1)    --返回 false，不相等
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Height

计算矩形的高度

```
Height ()
```

参数

无

返回值

类型	描述

number	矩形的高

注意事项

无

示例

```
rect = LXRect.new(10, 10, 50, 50)
nHeight = rect.Height()
print(nHeigth)        --输出 40
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Inflate

增加矩形的高和宽

```
Inflate (x, y)
Inflate(ptSize)
Inflate(l, t, r, b)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	左上点坐标向矩形中心增加的值	否	无
y	number	右下点坐标向矩形中心增加的值	否	无

参数名	类型	描述	是否可选	默认值
rect	LXRect	矩形以 rect 的四个值向矩形中心增大	否	无

参数名	类型	描述	是否可选	默认值
left	number	左上点 X 坐标增加值	否	无
top	number	左上点 Y 坐标增加值	否	无
right	number	右下点 X 坐标增加值	否	无
bottom	number	右下点 Y 坐标增加值	否	无

返回值

无

注意事项

无

示例

```
rect = LXRect.new(0,0,300,300)
rect.Inflate(50, 200)
--rect 的四个值变为-50,-200,350,500
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Intersect

矩形为二个指定的矩形的交叉部份

```
Intersect (srcRect1, srcRect2)
```

参数

参数名	类型	描述	是否可选	默认值
srcRect1	LXRect	源矩形 1	否	无
srcRect2	LXRect	源矩形 2	否	无

返回值

类型	描述
Boolean	操作是否成功

注意事项

无

示例

```
srcRect1 = LXRect.new(125, 0, 150, 200)
srcRect2 = LXRect.new(0, 75, 350, 95)
InterRect = LXRect.new()
bRes = InterRect.Intersect(srcRect1, srcRect2)
--返回 true，InterRect 的值为 125,75,150,95
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

IsEmpty

判断矩形是否为空

IsEmpty ()

参数

无

返回值

类型	描述
Boolean	矩形是否为空

注意事项

如果矩形的高和宽为 0 或者负值时，函数才返回 true

示例

```
Rect1 = LXRect.new(125, 0, 150, 200)
bRes = Rect1.IsEmpty()    --返回 false
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

IsNull

判断矩形左上角和右下角的四个值是否都为 0

IsNull ()

参数

无

返回值

类型	描述
Boolean	矩形四个值是否都为 0

注意事项

无

示例

```
Rect1 = LXRect.new(125, 0, 150, 200)
bRes = Rect1.IsNull()    --返回 false
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

MoveToX

将矩形向左移动指定值

```
MoveToX (x)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	向左移动值	否	无

返回值

无

注意事项

参数 x 如果为负值则取其绝对值进行操作，即 MoveToX(10)与 MoveToX(-10)效果相同

示例

```
Rect1 = LXRect.new(0, 0, 100, 100)
Rect1.MoveToX(10)     --Rect1 的四个值变为 10,0,110,100
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

MoveToXY

将矩形向左下方向移动指定值的正值

```
MoveToXY (x, y)
MoveToXY(Pt)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	向左移动值	否	无
y	number	向下移动值	否	无

参数名	类型	描述	是否可选	默认值
Pt	LXPoint	向左下移动值	否	无

返回值

```
无
```

注意事项

```
参看 MoveToX
```

示例

```
Rect1 = LXRect.new(0, 0, 100, 100)
Rect1.MoveToXY(10,10)      --Rect1 的四个值变为 10,10,110,110
```

参见

```
Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua
```

MoveToY

将矩形向下方向移动指定值的正值

```
MoveToY (y)
```

参数

参数名	类型	描述	是否可选	默认值
y	number	向下移动值	否	无

返回值

```
无
```

注意事项

参看 MoveToX

示例

```
Rect1 = LXRect.new(0, 0, 100, 100)
Rect1.MoveToY(10)      --Rect1 的四个值变为 0,10,100,110
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Normalize

规格化矩形使其高和宽都为正值

```
Normalize()
```

参数

无

返回值

无

注意事项

无

示例

```
rect1 = LXRect.new(110, 100, 250, 310);
rect2 = LXRect.new(250, 310, 110, 100);

rect1.Normalize();
rect2.Normalize();

bEqual = rect1.Equal(rect2)          --返回 true
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Offset

通过指定的值移动矩形

```
Offset(Pt)
Offset(x, y)
```

参数

参数名	类型	描述	是否可选	默认值
Pt	LXPoint	移动值	否	无

参数名	类型	描述	是否可选	默认值
x	number	X 坐标移动值	否	无
y	number	Y 坐标移动值	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
rect1 = LXRect.new(0, 0, 35, 35);
rect2 = LXRect.new(230, 230, 265, 265);

rect1.OffsetRect(230, 230);

bEqual = rect1.Equal(rect2)          --返回 true
```

参见

```
Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua
```

PtInRect

判断指定的点是否在矩形中

```
PtInRect (Pt)
```

参数

参数名	类型	描述	是否可选	默认值
Pt	LXPoint	指定点	否	无

返回值

类型	描述
Boolean	指定点是否在矩形中

注意事项

无

示例

```
pt1 = LXPoint.new(35,50)
pt2 = LXPoint.new(125,298)
rect = LXRect.new(5,5,100,100)
bIn = rect.PtInRect(pt1)          --返回 true
bIn = rect.PtInRect(pt2)          --返回 false
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

SetValue

设置矩形

```
SetValue(left, top, right, bottom)
```

参数

参数名	类型	描述	是否可选	默认值
left	number	左坐标	否	无
top	number	上坐标	否	无
right	number	右坐标	否	无
bottom	number	下坐标	否	无

返回值

无

注意事项

无

示例

```
rect = LXRect.new()
rect.SetValue(0,20,3,50)          --rect 的值变为 0,20,3,50
```

参见

```
Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua
```

SetEmpty

设置矩形为空矩形 (左上角和右下角的四个值都为 0)

```
SetEmpty ()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
rect = LXRect.new()
rect.SetValue(0,20,3,50)          --rect 的值变为 0,20,3,50
rect.SetEmpty()                      --rect 的值重新变为 0,0,0,0
```

参见

```
Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua
```

Size

计算矩形的宽和高

```
Size ()
```

参数

```
无
```

返回值

类型	描述
nWidth	矩形的宽
nHeight	矩形的高

注意事项

无

示例

```
rect = LXRect.new(5,5,100,200)
nWidth,nHeight = rect.Size()
print(nWidth)                --输出 95
print(nHeight)               --输出 195
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Subtract

矩形为二指定矩形相减后的矩形

```
Subtract (RectSrc1, RectSrc2)
```

参数

参数名	类型	描述	是否可选	默认值
srcRect1	LXRect	源矩形 1	否	无
srcRect2	LXRect	源矩形 2	否	无

返回值

类型	描述
Boolean	操作是否成功

注意事项

无

示例

```
rect1 = LXRect.new(10,10,100,100)
rect2 = LXRect.new(50,10,150,150)
rect = LXRect.new()
rect.Subtract(rect1, rect2)    --rect 为 rect1 减去 rect2 得到的矩形，值为 10,10,50,100
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

TopLeft

返回矩形左上角 left, top 二个值

```
TopLeft ()
```

参数

无

返回值

类型	描述
LXPoint	左上角坐标

注意事项

无

示例

```
rect1 = LXRect.new(10,10,100,100)
ptTopLeft =rect1.TopLeft()
print(ptTopLeft.GetX())          --输出 10
print(ptTopLeft.GetY())          --输出 10
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Union

矩形为指定二矩形的联合后的矩形

```
Union (RectSrc1, RectSrc2)
```

参数

参数名	类型	描述	是否可选	默认值
srcRect1	LXRect	源矩形 1	否	无
srcRect2	LXRect	源矩形 2	否	无

返回值

类型	描述
Boolean	操作执行后的矩形是否有效（不为空）

注意事项

无

示例

```
rectSrc1 = LXRect.new(100,0,200,300)
rectSrc2 = LXRect.new(0,100,300,200)
rect = LXRect.new()
rect.Union(rectSrc1, rectSrc2)          --rect 的值为 0,0,300,300
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

Width

计算矩形的宽度

```
Width ()
```

参数

无

返回值

类型	描述
number	矩形的宽

注意事项

无

示例

```
rect = LXRect.new(0,100,300,200)
print(rect.Width())        --输出 300
```

参见

Samples\Test_LuaBase\Test_LXRect\脚本\XMain.lua

LXRgn

功能描述

该对象实现对各种区域的操作。

函数列表

函数名称	说明
new	创建一个 LXRgn 对象
Combine	区域的联合
Copy	区域的拷贝
CreateFromBmp	从一 BMP 转化为相应的区域
CreateEllipse	创建一个椭圆区域
CreatePolygon	创建一个多边形
CreateRect	创建一个矩形区域
EqualRgn	区域是否相等
OffsetRgn	区域偏移
PtInRegion	鼠标点是否在区域内
RectInRegion	矩形是否在区域内
GetRgnBox	得到区域的边界矩形

new

新建区域对象

```
new()
```

参数

无

返回值

类型	描述
LXRgn	区域对象

注意事项

无

示例

```
region = LXRgn.new()
```

参见

S Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua

CreateFromBmp

从一 BMP 文件转化为相应的区域

```
CreateFromBmp (BmpPath, Left, Top)
```

参数

参数名	类型	描述	是否可选	默认值
BmpPath	string	图片的路径	否	无
Left	number	透明色位置 X 坐标	是	0
Top	number	透明色位置 Y 坐标	是	0

返回值

类型	描述
boolean	创建是否成功

注意事项

图像路径是相对路径，相对于应用程序所在目录，比如应用程序位于 D:\Xfun\ 目录下，则图片相对路径为”image.bmp”，绝对路径为”D:\Xfun\image.bmp”
取图像 Left Top 位置的颜色值为透明色
Left=0 Top=0 左上角 默认值
Left=-1 Top=0 右上角
Left=0 Top=-1 左下角
Left=-1 Top=-1 右下角

示例

```
rgn = LXRgn.new()
bRes = rgn.CreateFromBmp("..\\资源\\1.bmp")
if bRes == true then
```

```
    print("CreateFromBmp Success~~~")
else
    print("CreateFromBmp Failed~~~~")
end
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

CreateEllipse

创建一个椭圆区域

```
CreateEllipse (left, top, right, bottom)
CreateEllipse (lxRect)
```

参数

参数名	类型	描述	是否可选	默认值
left	number	椭圆矩形边界左坐标	否	无
top	number	椭圆矩形边界上坐标	否	无
right	number	椭圆矩形边界右坐标	否	无
bottom	number	椭圆矩形边界下坐标	否	无

参数名	类型	描述	是否可选	默认值
lxRect	LXRect	矩形边界	否	无

返回值

类型	描述
boolean	创建是否成功

注意事项

```
无
```

示例

```
region = LXRgn.new()
```

```
rect = LXRect.new(100, 100, 200, 200);
bRes = region.CreateEllipse(rect);
if bRes == true then
	print("CreateEllipse Success~~~")
end
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

CreatePolygon

创建一个多边形

```
CreatePolygon (TablePoint, Mode，  Count)
```

参数

参数名	类型	描述	是否可选	默认值
TablePoint	table	多边形顶点集合	否	无
Mode	string	创建模式	是	“ALTERNATE”
Count	number	绘制点个数	是	-1

返回值

类型	描述
boolean	创建是否成功

注意事项

```
Count 默认值为-1，表示绘制所有点
Mode 为创建的模式   有以下两种模式：“ALTERNATE”（默认值），“WINDING”
```

示例

```
rgn = LXRgn.new()
vptlist = {}
vptlist[1] = LXPoint.new(50, 50)
vptlist[2] = LXPoint.new(50, 100)
vptlist[3] = LXPoint.new(80, 60)
vptlist[4] = LXPoint.new(120, 240)
vptlist[5] = LXPoint.new(250, 100)
bRes = rgn.CreatePolygon(vptlist, 5, "ALTERNATE")
if bRes == true then
```

```
    print("CreateEllipse Success~~~")
end
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

CreateRect

创建一个矩形区域

```
CreateRect (Left, Top, Right, Bottom)
CreateRect (lxRect)
```

参数

参数名	类型	描述	是否可选	默认值
left	number	矩形左坐标	否	无
top	number	矩形上坐标	否	无
right	number	矩形右坐标	否	无
bottom	number	矩形下坐标	否	无
参数名	类型	描述	是否可选	默认值
lxRect	LXRect	矩形边界	否	无

返回值

类型	描述
boolean	创建是否成功

注意事项

```
无
```

示例

```
region1 = LXRgn.new()
region1.CreateRect(LXRect.new(10,10,100,100))
if bRes == true then
    print("Point In Region~~~")
end
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

Combine

区域的联合

```
Combine (lxRgn1, lxRgn2, CombineMode)
```

参数

参数名	类型	描述	是否可选	默认值
lxRgn1	LXRgn	区域 1	否	无
lxRgn2	LXRgn	区域 2	否	无
CombineMode	string	联合方式	否	无

返回值

类型	描述
number	联合结果

注意事项

```
CombineMode 为区域联合的方式　有以下几种方式
"RGN_AND"运用两个区域的相交区域
"RGN_COPY"拷贝 lxRgn1 的区域
"RGN_DIFF"区域包含 lxRgn1　但不包含 lxRgn2
"RGN_OR"联合两块区域
"RGN_XOR"联合两块区域　但去掉重叠区域

返回值取值如下:
= 0　联合失败
= 1　联合的新区域为空
= 2　新区域的没有边界
= 3　新区域的边界重叠
```

示例

```
region1 = LXRgn.new()
region1.CreateRect(50,50,150,150)
region2 = LXRgn.new()
region2.CreateRect(100,100,200,200)
region3 = LXRgn.new()
region3.CreateRect(0,0,50,50)
Res = region3.Combine(region2, region1, “RGN_AND”)
--region3 区域变为(100,100,150,150)
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

Copy

区域的拷贝

```
Copy (lxRgn1)
```

参数

参数名	类型	描述	是否可选	默认值
lxRgn1	LXRgn	需要拷贝的 Rgn	否	无

返回值

类型	描述
number	拷贝之后区域的类型

注意事项

```
返回值取值如下:
= 0        错误，不是有效的区域句柄
= 1        一个空的区域
= 2        没有重叠边框的区域
= 3        有重叠边框的区域
```

示例

```
region4 = LXRgn.new()
region2 = LXRgn.new()
region2.CreateRect(200, 0, 500, 200)
region4.Copy(region2)          --region4 与 region2 相同
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

EqualRgn

区域是否相等

```
EqualRgn (lxRgn1)
```

参数

参数名	类型	描述	是否可选	默认值
lxRgn1	LXRgn	判断是否相等的Rgn	否	无

返回值

类型	描述
boolean	是否相等

注意事项

无

示例

```
region1 = LXRgn.new()
region1.CreateRect(300, 0, 600, 200)
region2 = LXRgn.new()
region2.CreateRect(300, 0, 600, 200)
BeInRgn = region1.EqualRgn(region2)      -- 返回 true
```

参见

Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua

OffsetRgn

区域偏移

```
OffsetRgn (OffsetX, OffsetY)
```

参数

参数名	类型	描述	是否可选	默认值
OffsetX	number	偏移量 X	否	无
OffsetY	number	偏移量 Y	否	无

参数名	类型	描述	是否可选	默认值
lxpoint	LXPoint	偏移量	否	无

返回值

类型	描述
number	偏移之后区域的类型

注意事项

返回值取值如下:
= 0　　错误，不是有效的区域句柄
= 1　　一个空的区域
= 2　　没有重叠边框的区域
= 3　　有重叠边框的区域

示例

```
region5 = LXRgn.new()
region5.CreateRect(300, 0, 600, 200)　　--区域(300,0,600,200)
region5.OffsetRgn(-100, 0)　　--区域(200,0,500,200)
```

参见

Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua

PtInRegion

判断给定点是否在区域内

```
PtInRegion (lxPoint)
```

参数

参数名	类型	描述	是否可选	默认值
lxpoint	LXPoint	给定点	否	无

返回值

类型	描述
boolean	点是否在区域内

注意事项

无

示例

```
--判断鼠标点击位置是否在(0,0,100,100)这个矩形内
function OnLButtonDown(this, nEventID, SendName, param)
point = LXPoint.new(param.x, param.y)
region = LXRgn.new()
region.CreateRect(0,0,100,100)
BeInRgn = region.PtInRegion(point)
end
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

RectInRegion

判断给定区域是否在区域内

```
RectInRegion (lxRect)
```

参数

```
lxRect 为需要判断的矩形
```

返回值

类型	描述
boolean	矩形是否在区域内

注意事项

```
无
```

示例

```
rect = LXRect.new(10, 10, 20, 20)
region = LXRgn.new()
region.CreateRect(0,0,100,100)
BeInRgn = region.RectInRegion(rect)
print(BeInRgn)                  --输出 true
```

参见

```
Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua
```

GetRgnBox

得到区域的边界矩形

GetRgnBox (lxRect)

参数

无

返回值

类型	描述
LXRect	边界矩形

注意事项

无

示例

```
region = LXRgn.new()
region.CreateRect(0,0,100,100)
rect = region.GetRgnBox()
print(rect.left)        --输出 0
print(rect.top)             --输出 0
print(rect.right)           --输出 100
print(rect.bottom)          --输出 100
```

参见

Samples\Test_LuaBase\Test_LXRgn\脚本\XMain.lua

LXColor

功能描述

该对象实现对颜色的设置及其各种操作。

属性列表

属性名称	说明
r	颜色对象 r 值
g	颜色对象 g 值
b	颜色对象 b 值
a	颜色对象 Alpha 值

示例

```
clr = LXColor.new(255, 0, 255)
local r = clr.r    --r 为 255
local g = clr.g    --g 为 0
clr.b = 255    --clr 对象的 b 值为 255
clr.a = 0        --clr 对象的 a 值为 0
```

函数列表

函数名称	说明
new	创建一个 LXColor 对象
SetValue	设置颜色值
GetValue	获取颜色值
AdjustContrast	调整颜色的对比度
AdjustSaturation	调整颜色的饱和度
Lerp	使用颜色线性插值
Modulate	执行颜色的乘积混合
Negative	获取指定颜色的反向颜色
Scale	缩放指定颜色值
LimitRange	限制颜色值在指定范围
IsSame	该颜色是否与指定颜色近似相等

new

创建一个 LXColor 对象

```
new ()
new(r, g, b, a)
```

参数

参数名	类型	描述	是否可选	默认值
r	number	颜色 r 值	是	0
g	number	颜色 g 值	是	0
b	number	颜色 b 值	是	0
a	number	颜色 Alpha 值	是	255

方法 1: 创建一个默认的 LXColor 对象
方法 2: 创建一个指定的 rgba 的 LXColor 对象
r, g, b, a 都为 number 类型,范围值在 0 – 255 之间, 分别表示颜色的 rgba

值

返回值

类型	描述
LXColor	返回 LXColor 对象

注意事项

无

示例

```
local clr = LXColor.new(255,30,10,100)
```

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

SetValue

设置颜色值

```
SetValue(r, g, b, a)
```

参数

参数名	类型	描述	是否可选	默认值
r	number	颜色 R 值	否	无
g	number	颜色 G 值	是	0
b	number	颜色 B 值	是	0
a	number	颜色 Alpha 值	是	0

返回值

无

注意事项

无

示例

```
local clr = LXColor.new(0,255,0,0)
clr.SetValue(255,30,14,255)
```

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

GetValue

获取颜色值

GetValue ()

参数

无

返回值

类型	描述
number	颜色 r 值
number	颜色 g 值
number	颜色 b 值
number	颜色 Alpha 值

注意事项

无

示例

```
local clr = LXColor.new(152,5,6,255)
local r,g,b,a = clr.GetColor() –r=152, g=5, b=6, a=255
```

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

AdjustContrast

调整颜色的对比度

```
AdjustContrast (c)
AdjustContrast(clrSrc, c)
```

参数

参数名	类型	描述	是否可选	默认值
c	number	输出因子	否	无

参数名	类型	描述	是否可选	默认值
clrSrc	LXColor	对比颜色	否	无
c	number	输出因子	否	无

返回值

无

注意事项

各颜色分量按照公式输出
pOut->r = 0.5f + c * (pIn->r - 0.5f);　不改变 alpha

示例

```
local clr = LXColor.new(255, 0, 255 ,255)
clr. AdjustContrast(20.0)
```

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

AdjustSaturation

调整颜色的饱和度

```
AdjustSaturation (s)
AdjustSaturation(clrSrc, s)
```

参数

参数名	类型	描述	是否可选	默认值
s	number	标记	否	无

参数名	类型	描述	是否可选	默认值
clrSrc	LXColor	对比颜色	否	无
s	number	标记	否	无

返回值

无

注意事项

s 取值如下：
= 0　　颜色变为灰度
= 1　　保持原颜色
调整不修改 Alpha 值

示例

```
local clr = LXColor.new(255, 0, 255 ,255)
clr. AdjustSaturation(10.0)
```

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

Lerp

使用颜色线性插值

```
Lerp (clr, s)
Lerp(clr1, clr2, s)
```

参数

参数名	类型	描述	是否可选	默认值
clr	LXColor	颜色值	否	无
s	number	因子	否	无

参数名	类型	描述	是否可选	默认值
clr1	LXColor	颜色值 1	否	无
clr2	LXColor	颜色值 2	否	无
s	number	因子	否	无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

Modulate

执行颜色的乘积混合

```
Modulate (clr, s)
Modulate(clr1, clr2 ,s)
```

参数

参数名	类型	描述	是否可选	默认值
clr	LXColor	颜色值	否	无
s	number	因子	否	无

参数名	类型	描述	是否可选	默认值
clr1	LXColor	颜色值 1	否	无
clr2	LXColor	颜色值 2	否	无
s	number	因子	否	无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

Negative

获取自身颜色或指定颜色的反向颜色，不改变 alpha 值

```
Negative ()
Negative(clr)
```

参数

参数名	类型	描述	是否可选	默认值
clr	LXColor	指定颜色值	否	无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

Scale

缩放指定颜色值

```
Scale ( s)
Scale(clr, s）
```

参数

参数名	类型	描述	是否可选	默认值
clr	LXColor	颜色值	否	无
s	number	因子	否	无

参数名	类型	描述	是否可选	默认值
clr1	LXColor	颜色值 1	否	无
clr2	LXColor	颜色值 2	否	无
s	number	因子	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

LimitRange

限制颜色值在指定范围

LimitRange ()

参数

无

返回值

无

注意事项

无

示例

Clr. LimitRange()

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

IsSame

该颜色是否与指定颜色近似相等

IsSame (clr, fError)

参数

参数名	类型	描述	是否可选	默认值
clr	LXColor	指定颜色值	否	无
fError	number	容差	否	无

返回值

类型	描述
Boolean	是否近似相等

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXColor\脚本\XMain.lua

LXFont

功能描述

该对象实现系统字体的创建、设置、属性获取以及文本绘制。

函数列表

函数名称	说明
new	创建字体绘制对象
Create	创建字体
GetDefaultFont	获得系统默认字体
SetCurFont	设置为当前字体
RestoreFont	还原先前字体
TextOut	单行绘制文本
DrawText	矩形区绘制文本
GetName	得到字体名
GetSize	得到字体大小
GetHeight	得到字体高度
GetMaxWidth	得到字体最大宽度
IsBold	是否为粗体
IsItalic	是否为斜体
IsUnderline	是否有下划线
IsCurFont	是否为当前系统字体

new

创建字体绘制对象

```
new()
```

参数

无

返回值

类型	描述
LXFont	LXFont 对象

注意事项

无

示例

```
local font = LXFont.new()     -- 创建一个 LXFont 对象
```

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

Create

创建字体

```
Create(nSize， strName, bBold, bTtalic, bUnderline, strCharSet)
```

参数

参数名	类型	描述	是否可选	默认值
nSize	number	字体大小	是	12
strName	string	字体名	是	nil
bBold	Boolean	是否粗体	是	false
bItalic	Boolean	是否斜体	是	false
bUnderline	Boolean	是否下划线	是	false
strCharSet	string	字符集	是	“DEFAULT”

strCharSet 取下列字符值之一:
ANSI
BALTIC
CHINESEBIG5

```
DEFAULT
EASTEUROPE
GB2312
GREEK
HANGUL
MAC
OEM
RUSSIAN
SHIFTJIS
SYMBOL
TURKISH
VIETNAMESE
OHAB
ARABIC
HEBREW
THAI
OEM
```

返回值

类型	描述
Boolean	是否创建成功

注意事项

```
无
```

示例

```
local font = LXFont.new()
font.Create(10)
font.Create(10,”宋体”)
font.Create(20,”宋体”,true,false,true)
```

参见

```
Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua
```

GetDefaultFont

获得系统默认字体

```
GetDefaultFont()
```

参数

无

返回值

类型	描述
Boolean	是否获取成功

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

SetCurFont

设置为当前字体

```
SetCurFont()
```

参数

无

返回值

无

注意事项

设置完后可用 IsCurFont()检查是否为当前字体

示例

```
local font = LXFont.new()
font.Create(18, “宋体”)
font.SetCurFont()                    --设为当前字体
bcurfont = font.IsCurFont()          --是否为当前字体
if bcurfont then
    print("font Set Success~~~")
end
font.RestoreFont()
```

参见

```
Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua
```

RestoreFont

还原先前字体

```
RestoreFont()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
在 SetCurFont()之后调用
```

示例

```
font.SetCurFont()                    --设为当前字体
bcurfont = font.IsCurFont()          --是否为当前字体
if bcurfont then
    print("font Set Success~~~")
end
font.RestoreFont()
```

参见

```
Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua
```

TextOut

单行绘制文字

```
TextOut(x, y, text, color)
TextOut(x, y, text, r, g, b)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	绘制起始横坐标	否	无
y	number	绘制起始纵坐标	否	无
text	string	绘制文字	否	无
color	LXColor	文字颜色	是	白色
r	number	颜色 R 值	否	0
g	number	颜色 G 值	否	0
b	number	颜色 B 值	否	0

返回值

类型	描述
Boolean	是否绘制成功

注意事项

无

示例

```
local font = LXFont.new()
font.Create(18, “宋体”)
font.SetCurFont()
font.TextOut(10,50,” 绘制文本”)
font.TextOut(0,100,”TextOut”,color)
```

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

DrawText

矩形区绘制文字

```
DrawText(rect, text, format, color)
DrawText(rect, text, format, r, g, b)
```

参见

参数名	类型	描述	是否可选	默认值
rect	LXRect	目标矩形区	否	无
text	string	绘制文字	否	无
format	string	文字格式	否	无
color	LXColor	文字颜色	是	白色
r	number	颜色 R 值	否	0
g	number	颜色 G 值	否	0
b	number	颜色 B 值	否	0

返回值

类型	描述
Boolean	是否绘制成功

注意事项

文字格式 format 支持以下格式：

上对齐	“t”
左对齐	“l”
右对齐	“r”
下对齐	“b”
居中	“c”
水平居中	“v”
单行	“s”
计算矩形区	“m”

示例

```
local font = LXFont.new()
font.Create(18, “宋体”)
font.SetCurFont()
font.DrawText (rect,”绘制文本”,”c”)
font.DrawText (rect,”aassvv”,”tl”)
font.DrawText (rect,”DrawText”,0,color)
```

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

GetName

得到字体名

```
GetName()
```

参数

无

返回值

类型	描述
string	字体名

注意事项

无

示例

```
local font = LXFont.new()
font.Create(20, "宋体", true, false, true)
name = font.GetName()
print(name)                                    --输出宋体
```

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

GetSize

得到字体大小

```
GetSize()
```

参数

无

返回值

类型	描述
number	字体大小

注意事项

无

示例

```
local font = LXFont.new()
font.Create(20, "宋体", true)
size = font.GetSize()                    --得到字体大小
print(size)                              --输出 9
```

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

GetHeight

得到字体高度

```
GetHeight()
```

参数

无

返回值

类型	描述
number	字体高度

注意事项

无

示例

```
local font = LXFont.new()
font.Create(20, "宋体")
height = font.GetHeight()                --得到字体高度
print(height)                            --输出 12
```

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

GetMaxWidth

得到字体最大宽度

```
GetMaxWidth()
```

参数

无

返回值

类型	描述
number	字体最大宽度

注意事项

无

示例

```
local font = LXFont.new()
font.Create(20, "宋体")
width = font.GetMaxWidth()          --得到字体最大宽度
print(width)                        --输出 12
```

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

IsBold

是否为粗体

```
IsBold()
```

参数

无

返回值

类型	描述
Boolean	是否为粗体

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

IsItalic

是否为斜体

IsItalic()

参数

无

返回值

类型	描述
Boolean	是否为斜体

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

IsUnderline

是否有下划线

IsUnderline()

参数

无

返回值

类型	描述
Boolean	是否有下划线

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

IsCurFont

是否为当前字体

```
IsCurFont()
```

参数

无

返回值

类型	描述
Boolean	是否为当前字体

注意事项

无

示例

无

参见

Samples\Test_LuaBase\Test_LXFont\脚本\XMain.lua

LXHandle

功能描述

该对象实现对系统资源的装载，常用系统资源有：图像，光标、纹理、材质、模型、动画集等。这些系统资源可以通过此类来装载。

函数列表

函数名称	说明
new	创建一个 LXHandle 对象，并加载相应的资源
GetResFileName	得到资源对象对应的文件路径名
Close	释放资源对象
IsValidate	检测资源是否有效
Replace	替换当前资源

new

以某种资源类型加载一个指定文件名的资源

```
new(strResName, strResType)
```

参数

参数名	类型	描述	是否可选	默认值
strResName	string	资源路径	否	无
strResType	string	资源类型	是,参考注意事项	nil

返回值

类型	描述
bool	加载成功返回 true ,加载失败返回 false

注意事项

strResName 既可以为相对路径(想对于该脚本)；也可以是一个绝对路径
strResType 可选，默认为空。其值有以下几种方式：

值	描述
"TEXTURE"	纹理
"MATERIAL"	材质
"MODEL"	模型

"ANIMATE"	动画集
"EFFECT"	效果文件
"CURSOR"	光标
"FONT"	字体
"BONE"	骨架
"SOUND"	声音
"IMAGE"	图像

示例

```
handle = LXHandle.new("..\\资源\\1.jpg")
--或者
handle = LXHandle.new("..\\资源\\1.jpg", "image")
```

参见

```
Samples\Test_LuaBase\Test_LXHandle\脚本\XMain.lua
```

GetResFileName

得到资源对象对应的文件路径名

```
GetResFileName()
```

参数

```
无
```

返回值

```
字符串型，资源的文件路径名
```

注意事项

```
无
```

示例

```
handle = LXHandle.new("..\\资源\\1.jpg", "image")
strResName = handle.GetResFileName()
print(strResName)                    --输出 1.jpg 的文件路径
```

参见

```
Samples\Test_LuaBase\Test_LXHandle\脚本\XMain.lua
```

Close

释放资源对象

```
Close()
```

参数

无

返回值

无

注意事项

使用资源完毕后，调用此方法来释放资源

示例

```
handle.Close()
```

参见

Samples\Test_LuaBase\Test_LXHandle\脚本\XMain.lua

IsValidate

检测资源是否有效

```
IsValidate()
```

参数

无

返回值

类型	描述
bool	资源有效返回 true,否则返回 false

注意事项

无

示例

```
bRes = handle.IsValidate()
if bRes then
	print("handle is Validate")
end
```

参见

```
Samples\Test_LuaBase\Test_LXHandle\脚本\XMain.lua
```

Replace

替换当前资源

```
Replace (strResName, strResType)
```

参数

参数名	类型	描述	是否可选	默认值
strResName	string	资源路径	否	无
strResType	string	资源类型	是,参考注意事项	nil

返回值

类型	描述
bool	资源有效返回 true,否则返回 false

注意事项

```
参见 new 函数
```

示例

```
--handle 代表 1.jpg 文件
local handle = LXHandle.new(“..\\资源\\1.jpg”)
--Replace 之后代表 2.jpg 文件
handle.Replace("..\\资源\\2.jpg")
```

参见

```
Samples\Test_LuaBase\Test_LXHandle\脚本\XMain.lua
```

LXPublic

功能描述

LXPublic 的提供了与系统相关的函数，用户可以通过这些函数获得计算机 CPU、内存、IE 版本号、系统时间、应用程序目录、屏幕分辨率等信息。如果用户的数据需要保存，可以使用存储读取配置文件的函数进行存储或读取。用户需要注意的一点，LXPublic 对象在创建非空项目时系统会自动生成，这时 XMain.lua 文件中会自动出现 SysFunc = LXPublic.new()

语句，通过 SysFunc 这个对象，用户可以直接使用 LXPublic 的所用成员函数。

函数列表

函数名称	说明
new	新建系统函数对象
GetMemoryStatus	获得内存信息
GetCPUInfo	获得 CPU 信息
GetOSVersion	获得操作系统版本号
GetIEVersion	获得 IE 版本号
GetIP	获得本机 IP 地址
GetComputerName	获得计算机名
OpenFile	打开配置文件
ReadKeyValue	读取配置文件键值
WriteKeyValue	写入配置文件键值
GetScreenDimension	获得屏幕分辨率，颜色深度
SetScreenDimension	设置屏幕分辨率，颜色深度
GetCurTime	获得当前系统时间
GetTickCount	获得系统启动毫秒数
GetHighTickCount	获得系统启动高精度时间
GetKeyState	获得按键状态
GetCurDir	获得当前应用程序路径
GetFontNameList	获得系统字体名称列表

new

新建系统函数对象

```
new()
```

参数

无

返回值

类型	描述
LXPublic	LXPublic 对象

注意事项

无

示例

```
Public = LXPublic.new()
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua
```

GetMemoryStatus

获得内存信息

```
GetMemoryStatus()
```

参数

```
无
```

返回值

类型	描述
table	内存信息列表

注意事项

```
返回的信息列表元素分别表示为：
"MemoryLoad"            内存使用百分比
"TotalPhys"             总物理内存大小，以字节(byte)为单位
"AvailPhys"             可用物理内存大小，以字节(byte)为单位
"TotalPageFile"         总页面文件大小，以字节(byte)为单位
"AvailPageFile"         可用页面文件大小，以字节(byte)为单位
"TotalVirtual"          总虚拟地址空间，以字节(byte)为单位
"AvailVirtual"          可用虚拟地址空间，以字节(byte)为单位
"AvailExtendedVirtual"
```

示例

```
memInfo = SysFunc.GetMemoryStatus ()
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua
```

GetCPUInfo

获得 CPU 信息

GetCPUInfo()

参数

无

返回值

类型	描述
table	CPU 信息列表

注意事项

返回的信息列表元素分别表示为：
“v_name“ CPU 厂商名如 GenuineIntel
“model_name” CPU 名称如 INTEL Pentium-III
“family” CPU 架构如 6 = Pentium-Pro 架构
“model”
“stepping” processor revision numberber
“feature” CPU 特征位
“os_support” 操作系统的支持的特征位
“checks” 已经对 OS 做过测试的特征掩码
“count” CPU 数量
"speed" CPU 频率

示例

```
cpuInfo = SysFunc.GetCPUInfo ()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetOSVersion

获得操作系统版本号

GetOSVersion()

参数

无

返回值

类型	描述

table	操作系统版本号列表

注意事项

返回的信息列表元素分别表示为：
"MajorVersion" 操作系统主版本号
"MinorVersion" 操作系统副版本号
"BuildNumber" The build number of the operating system
"PlatformId" 平台 ID

示例

```
osVersion = Public.GetOSVersion ()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetIEVersion

获得 IE 版本号

```
GetIEVersion()
```

参数

无

返回值

类型	描述
string	IE 版本号

注意事项

无

示例

```
Version = SysFunc.GetIEVersion()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetIP

获得本机 IP 地址

```
GetIP()
```

参数

无

返回值

类型	描述
string	本机 IP 地址

注意事项

无

示例

```
ip = SysFunc.GetIP ()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetComputerName

获得计算机名

```
GetComputerName()
```

参数

无

返回值

类型	描述
string	计算机名

注意事项

无

示例

```
name = SysFunc.GetComputerName ()
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua
```

OpenFile

打开配置文件

```
OpenFile(FileName,)
```

参数

参数名	类型	描述	是否可选	默认值
FileName	string	配置文件名	否	无

返回值

类型	描述
Boolean	操作是否成功

注意事项

```
若指定配置文件不存在，则自动创建
```

示例

```
bRes = SysFunc.OpenFile(“my.ini”)
if bRes then
    print(“OpenFile Success~~~~”)
end
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua
```

ReadKeyValue

读取配置文件键值

```
ReadKeyValue(section, key)
```

参数

参数名	类型	描述	是否可选	默认值
section	string	节点名	否	无
key	string	键名	否	无

返回值

类型	描述
string	键值

注意事项

无

示例

```
value = SysFunc.ReadKeyValue("Test","key1")
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

WriteKeyValue

写入配置文件键值

```
WriteKeyValue(section, key，value)
```

参数

参数名	类型	描述	是否可选	默认值
section	string	节点名	否	无
key	string	键名	否	无
value	string	写入的新键值	否	无

返回值

无

注意事项

无

示例

```
SysFunc.WriteKeyValue("Test","key2","NewValue")
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetScreenDimension

获得屏幕分辨率，颜色深度

GetScreenDimension ()

参数

无

返回值

类型	描述
number	屏幕像素宽度
number	屏幕像素高度
number	屏幕颜色深度

注意事项

无

示例

width,height,pixel = SysFunc.GetScreenDimension()

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

SetScreenDimension

设置屏幕分辨率，颜色深度

SetScreenDimension (width，height，pixel)

参数

参数名	类型	描述	是否可选	默认值
width	number	屏幕像素宽度	否	无
height	number	屏幕像素高度	否	无
pixel	number	屏幕颜色深度	否	无

返回值

无

注意事项

无

示例

```
SysFunc.SetScreenDimension(800,600,16)
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetCurTime

获得当前系统时间

```
GetCurTime ()
```

参数

无

返回值

类型	描述
table	当前系统时间列表

注意事项

返回信息列表中元素分别表示：
"year" 年
"month" 月
"day" 日
"dayofweek" 星期
"hour" 小时
"minute" 分钟
"second" 秒

示例

```
time = SysFunc.GetCurTime()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetTickCount

获得系统启动后消耗的毫秒数

```
GetTickCount()
```

参数

无

返回值

类型	描述
number	系统启动后消耗的毫秒数

注意事项

无

示例

```
count = SysFunc.GetTickCount()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetHighTickCount

获得系统启动后消耗的高精度时间

```
GetHighTickCount()
```

参数

无

返回值

类型	描述
number	系统启动后消耗的毫秒数

注意事项

无

示例

```
count = SysFunc.GetHighTickCount ()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

GetKeyState

获得按键状态

```
GetKeyState(virtkey)
```

参数

参数名	类型	描述	是否可选	默认值
virtkey	Number 或 String	虚拟键值	否	无

返回值

类型	描述
boolean	按键是否被按下
boolean	按键是否被激活

注意事项

常用的虚拟键值：

0 - 9	"0" - "9"
A - Z	"A" - "Z"
鼠标左键	1
鼠标右键	2
鼠标中键	4
Backspace	8
Tab	9
Enter	13
Shift	16
LShift	160
RShift	161
Control	17

Escape	27
LControl	162
RControl	163
Alt	18
LAlt	164
RAlt	165
Pause	19
Caps Lock	20
Space	32
Page up	33
Page down	34
End	35
Home	36
Left Arrow	37
Up Arrow	38
Right Arrow	39
Down Arrow	40
Select	41
Print Screen	44
Insert	45
Delete	46
LWin	91
RWin	92
Sleep	95
小键盘 0	96
小键盘 1	97
小键盘 2	98
小键盘 3	99
小键盘 4	100
小键盘 5	101
小键盘 6	102
小键盘 7	103
小键盘 8	104
小键盘 9	105
小键盘*	106
小键盘+	107
|	108
-	109
.	110
/	111
F1	112
F1 到 F11 依次加 1	
F12	123
number Lock	144

```
Scroll Lock		145
```

示例

```
press,enable = SysFunc.GetKeyState ("2")
press,enable = SysFunc.GetKeyState ("A")
press,enable = SysFunc.GetKeyState (162)
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua
```

GetCurDir

获得当前应用程序路径

```
GetCurDir()
```

参数

```
无
```

返回值

类型	描述
string	当前应用程序路径

注意事项

```
无
```

示例

```
dir = SysFunc.GetCurDir ()
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua
```

GetFontNameList

获得系统字体名称列表

```
GetFontNameList()
```

参数

无

返回值

类型	描述
table	系统字体名称列表

注意事项

无

示例

```
font = SysFunc.GetFontNameList ()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXPublic\脚本\XMain.lua

LXImageList

功能描述

实现图像列表

函数列表

函数名称	说明
new	创建一个 ImageList
Add	添加一幅图片
Remove	删除一副图片
Replace	替换一副图片
GetImageCount	得到图像列表包含的图像个数

new

创建一个 ImageList

```
new (cx，cy，Flags，nInitial，nGrow)
new(strFile, cx, nGrow, color)
```

参数

参数名	类型	描述	是否可选	默认值

cx	number	宽度	否	无
cy	number	高度	否	无
Flags	number	图像列表类型	否	无
nInitial	number	链表初始包含图像个数	否	无
nGrow	number	图像增长时预留空间个数	否	无
strFile	string	图像列表路径	否	无
Color	LXColor	掩码颜色	否	无

返回值

类型	描述
number	0 失败, 1 成功

注意事项

Flags 的类型
0 --如果没有其它标记被确定，则使用缺省行为。典型地，缺省为 4, ;但对于旧的显示驱动程序，缺省为 254
1 --使用掩码。图象列表包含两个位图，其中一个是用做掩码的位图。如果不包括此值，图象列表只包含一个位图
4 --使用 4 位（16 色）设备独立位图(DIB)部分作为图象列表的位图
8 --使用 8 位 DIB 部分。彩色表格使用的颜色与半色调调色板的一样
16 --使用 16 位（32/64K 色）DIB 部分
24 --使用 24 位 DIB 部分
32 --使用 32 位 DIB 部分
254 --使用设备独立位图

示例

```
ImageList. Create(32，32，32，6，4)
local path = GetXsnPath()
ImageList.Create(path .. “Image\\123.bmp”, 10, 4, LXColor.new(255, 255, 255, 255)
```

参见

Samples\Test_LuaBase\Test_LXImageList\脚本\XMain.lua

Add

添加一幅图片

```
Add(strFileName)
Add(objHand)
```

参数

参数名	类型	描述	是否可选	默认值
strFileName	string	图片路径	否	无
objHand	LXHandle	图片句柄	否	无

返回值

类型	描述
number	图像在链表中的索引

注意事项

返回值
-1 表示失败

示例

```
imagelist = LXImageList.new(64, 64, 32, 3, 2)
handle1 = LXHandle.new("..\\Image\\x.jpg");
imagelist.Add(handle1)
print(imagelist.GetImageCount())
imagelist.Add("..\\Image\\x.jpg");
imagelist.Add("..\\Image\\x.jpg");
imagelist.Add("..\\Image\\y.jpg");
imagelist.Add("..\\Image\\y.jpg");
imagelist.Add("..\\Image\\y.jpg");
```

参见

Samples\Test_LuaBase\Test_LXImageList\脚本\XMain.lua

Remove

删除一副图片

```
Remove (Index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	图像索引	否	无

返回值

类型	描述
Boolean	0 失败, 1 成功

注意事项

无

示例

```
imagelist.Remove(1)
```

参见

Samples\Test_LuaBase\Test_LXImageList\脚本\XMain.lua

Replace

替换一副图片

```
Replace(index, strFileName)
Replace(Index, objHandle)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	图片列表中的索引	否	无
strFileName	string	图片路径	否	无
ObjHandle	LXHandle	替换的图像句柄	否	无

返回值

类型	描述
number	0 表示失败, 1 表示成功

注意事项

Index 以 0 开始

示例

```
imagelist.Replace(1, "..\\Image\\y.jpg")
handle = LXHandle.new("..\\Image\\y.jpg")
imagelist.Replace(1, handle)
```

参见

Samples\Test_LuaBase\Test_LXImageList\脚本\XMain.lua

GetImageCount

得到图像列表包含的图像个数

GetImageCount()

参数

无

返回值

类型	描述
number	图像个数

注意事项

无

示例

Count = ImageList.GetImageCount()

参见

Samples\Test_LuaBase\Test_LXImageList\脚本\XMain.lua

LXCursor

功能描述

该对象实现对光标资源的装载、显示及操作。光标资源文件以”xcur”为扩展名，可为硬盘全路径文件名，也可为系统资源文件.

函数列表

函数名称	说明
new	创建一个 LXCursor 对象
Load	加载一个指定文件名的资源
GetResFileName	得到资源对象对应的文件路径名
Close	释放资源对象
IsValidate	检测资源是否有效

new

创建一个新的光标对象

```
new(strCursorFile)
```

参数

参数名	类型	描述	是否可选	默认值
strCursorFile	string	光标文件名 或系统资源光标	否	无

返回值

类型	描述
LXCursor	LXCursor 对象表示：系统光标对象

注意事项

```
注: 光标资源文件以”xcur”为扩展名
系统中自动拥有如下默认光标:
"Arrow"          //默认箭头光标
"Edit"           //默认 I 光标
"hand"           //默认手型光标
"lrArrow"        //默认<-->光标
"udArrow"        //默认|光标
"ldArrow"        //默认/光标
"rdArrow"        //默认\光标
"busy"           //默认忙光标。
```

示例

```
创建一个系统光标
cur =LXCursor.new ("busy")
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXCursor\脚本\XMain.lua

SetCursor

设置自己为系统当前光标

```
SetCursor()
```

参数

无

返回值

无

注意事项

无

示例

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXCursor\脚本\XMain.lua

SetCursorPos

设置当前光标位置

```
SetCursorPos(nX ,  nY )
```

参数

参数名	类型	描述	是否可选	默认值
nX	number	横坐标	否	无
nY	number	纵坐标	否	否

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXCursor\脚本\XMain.lua

GetHotPos

获取光标热点位置

GetHotPos()

参数

无

返回值

类型	描述
number	表示热点横坐标
number	表示热点纵坐标

注意事项

无

示例

无

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXCursor\脚本\XMain.lua

GetAniCount

获取光标帧数

GetAniCount()

参数

无

返回值

类型	描述
number	表示光标动画桢数

注意事项

示例

```
Count = cur.GetAniCount()
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LXCursor\脚本\XMain.lua
```

CloseCursor

关闭当前光标

```
CloseCursor ()
```

参数

```
无
```

返回值

```
无
```

注意事项

示例

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LXCursor\脚本\XMain.lua
```

3.4.2 图像与声音

本部分介绍了图像声音导出类，导出类名称及功能如下表所示

对象名	功能
LX2DShape	2D 图像绘制，用于绘制点，线，面等 2 维图像
LXTexture	纹理对象，用于将指定的纹理文件绘制到屏幕上
LXSound	声音对象，用于播放指定的声音文件
LXImage	内存位图对象，用于在内存中进行图像处理

如果用户需要在屏幕上绘制简单的 2 维图形，可以首先调用 LX2DShape 导出类的 new 函数新建一个 LX2DShape 导出类的对象，然后调用它的成员函数进行绘制，可以绘制的图形包括：点、直线、矩形、多边形、圆\椭圆、圆角矩形、弧、弦。对于矩形、多边形、圆\椭圆 、圆角矩形、弦，还可以调用函数以指定的颜色填充图形内部区域。

如果用户需要在屏幕上绘制纹理，可以通过一个纹理的句柄 LXHandle 对象或者文件的绝对或相对路径创建一个 LXTexture 的对象，并调用它的绘制成员函数进行纹理绘制，如果用户需要对已创建的纹理对象中某一帧的图像内容进行修改，首先需要通过 LXTexture 的导出函数获得纹理中指定帧的内存位图，然后调用 LXImage 的导出函数进行图像数据的修改，保存操作。

如果用户需要在自己的工程中加入声音文件，可以通过一个声音文件的绝对路径或相对路径创建一个 LXSound 对象，并调用它的成员函数进行播放控制。

LX2Dshape

功能描述

该对象实现对二维图像的基本绘制和各种多边形的填充.包括绘制点、直线、矩形、角矩形、圆\椭圆、填充矩形圆角矩形椭圆等操作。

函数列表

函数名称	说明
new	创建一个 LX2Dshape 对象
DrawPoint	绘制一个点
DrawLine	绘制一条线段
DrawRect	绘制一矩形
DrawPolygon	绘制多边形
FillRect	填充矩形
FillPolygon	填充多边形
MoveTo	将画笔移动到指定点
LineTo	绘制到指定的点
DrawEllipse	绘制椭圆
FillEllipse	填充椭圆
DrawChord	绘制弦

FillChord	填充弦
DrawRoundRect	绘制圆角矩形
FillRoundRect	填充圆角矩形
DrawArc	绘制弧
SetViewPortBKColor	设置当前视区的背景颜色
GetViewPortBKColor	获取当前视区的背景颜色

new

创建一个新的 LX2Dshape 对象

```
new()
```

参数

无

返回值

类型	描述
LX2DShape	LX2DShape 对象

注意事项

无

示例

```
shape = LX2DShape.new()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

DrawPoint

绘制一个点

```
DrawPoint(nx, ny, clr)
DrawPoint(Pt,clr)
```

参数

参数名	类型	描述	是否可选	默认值
nx	number	X 坐标	否	无
ny	number	Y 坐标	否	无

clr	LXColor	点颜色	否	无

参数名	类型	描述	是否可选	默认值
Pt	LXVector2	点坐标	否	无
clr	LXColor	点颜色	否	无

返回值

类型	描述
boolean	是否绘制成功

注意事项

无

示例

```
shape.DrawPoint(50,50，LXColor.new(255,0,0, 255))
shape.DrawPoint(LXVector2.new(50,50)，LXColor.new(255,0,0, 255))
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

DrawLine

绘制一条线段

```
DrawLine (PtStart, PtEnd, clr, dWidth, dPattern, dPatternScale)
```

参数

参数名	类型	描述	是否可选	默认值
PtStart	LXVector2	线段起始点坐标	否	无
PtEnd	LXVector2	线段终点坐标	否	无
clr	LXColor	线段颜色	否	无
dWidth	number	线宽	是	1
dPattern	number	线样式	是	-1
dPatternScale	number	线缩放因子	是	1

返回值

类型	描述
boolean	是否绘制成功

注意事项

线宽 dWidth 默认值为 1
线样式 dPattern 默认值为-1
线缩放因子 dPatternScale 默认值为 1

示例

```
PtStart = LXVector2.new(100,150)
PtEnd = LXVector2.new(150,180)
shape.DrawLine(PtStart, PtEnd, LXColor.new(255, 0, 0, 255))
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

DrawRect

绘制一矩形

```
DrawRect(rect, clr)
DrawRect(pt1, pt2, clr)
DrawRect(l, t, r, b, clr)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	需要绘制的矩形	否	无
pt1	LXVector2	矩形左上角的点	否	无
pt2	LXVector2	矩形右下角的点	否	无
l	number	矩形左上角 x 坐标	否	无
t	number	矩形左上角 y 坐标	否	无
r	number	矩形右下角 x 坐标	否	无
b	number	矩形右下角 y 坐标	否	无
clr	LXColor	矩形颜色	否	无

返回值

类型	描述
boolean	是否绘制成功

注意事项

无

示例

```
rect = LXRect.new(100,100,300,400)
Clr = LXColor.new(255, 0, 0, 255)
shape.DrawRect(rect, Clr)
shape.DrawRect(100, 100, 300, 400,Clr)
Pt1 = LXVector2.new(100, 100)
Pt2 = LXVector2.new(300, 400)
shape.DrawRect(Pt1, Pt2, Clr)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

DrawPolygon

绘制多边形

```
DrawPolygon (Pts, LXColor, nCount, dWidth, dPattern, dPatternScale)
```

参数

参数名	类型	描述	是否可选	默认值
Pts	table	多边形顶点坐标集合	否	无
clr	LXColor	直线颜色	否	无
nCount	number	要绘制点的个数	是	-1
dWidth	number	线宽	是	1
dPattern	number	线样式	是	-1
dPatternScale	number	线缩放因子	是	1

返回值

类型	描述
boolean	是否绘制成功

注意事项

nCount 默认值为-1，表示绘制 Pts 中的所有点
dWidth 默认值为 1
dPattern 默认值为-1
dPatternScale 默认值为 1

示例

```
clr = LXColor.new(255,0,200,255)
tt = {}
for i =1,10,1 do
	pt = LXVector2.new(50+i*5,150-i*5)
tt[i] = pt
end
shape = LX2DShape.new()
shape.DrawPolygon(tt,clr)
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua
```

FillRect

填充矩形

```
FillRect(rect, clr)
FillRect(pt1, pt2, clr)
FillRect(l, t, r, b, clr)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	需要绘制的矩形	否	无
pt1	LXVector2	矩形左上角的点	否	无
pt2	LXVector2	矩形右下角的点	否	无
l	number	矩形左上角 x 坐标	否	无
t	number	矩形左上角 y 坐标	否	无
r	number	矩形右下角 x 坐标	否	无
b	number	矩形右下角 y 坐标	否	无
clr	LXColor	矩形颜色	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
clr = LXColor.new(255,0,200,255)
rect = LXRect.new(100,100,400,400)
shape = LX2DShape.new()
```

```
shape.FillRect(rect,clr)
shape.FillRect(100, 100, 400, 400,clr)
Pt1 = LXVector2.new(100, 100)
Pt2 = LXVector2.new(300, 400)
shape.FillRect(Pt1, Pt2, clr)
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua
```

FillPolygon

填充多边形

```
FillPolygon(Pts, clr, nCount)
```

参数

参数名	类型	描述	是否可选	默认值
Pts	LXRect	需要绘制的矩形	否	无
clr	LXColor	矩形颜色	否	无
nCount	number	要绘制的点个数	是	-1

返回值

类型	描述
number	绘制是否成功, 0 失败, 1 成功

注意事项

```
nCount 默认值为-1，表示绘制 Pts 中的所有点
```

示例

```
clr = LXColor.new(255,0,200,255)
tt = {}
for i =1,10,1 do
	pt = LXVector2.new(50+i*5,150-i*5)
tt[i] = pt
end
shape = LX2DShape.new()
shape.FillPolygon(tt, clr, 5)
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua
```

MoveTo

将画笔移动到指定点

```
MoveTo(x, y)
MoveTo(Pt)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	X 坐标	否	无
y	number	Y 坐标	否	无

参数名	类型	描述	是否可选	默认值
Pt	LXVector2	坐标点	否	无

返回值

无

注意事项

无

示例

```
shape.MoveTo(50,50)
shape.MoveTo(LXVector2.new(50,50))
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

LineTo

绘制到指定的点

```
LineTo(EndPt,clr, dWidth, dPattern, dPatternScale)
LineTo(EndX, EndY, clr, dWidth, dPattern, dPatternScale)
```

参数

参数名	类型	描述	是否可选	默认值
EndPt	LXVector2	坐标点	否	无

EndX	number	绘制点 X 坐标	否	无
EndY	number	绘制点 Y 坐标	否	无
clr	LXColor	颜色值	否	无
dWidth	number	线宽	是	1
dPattern	number	线样式	是	-1
dPatternScale	number	线缩放因子	是	1

返回值

类型	描述
boolean	是否绘制成功

注意事项

dWidth 默认值为 1
dPattern 默认值为-1
dPatternScale 默认值为 1

示例

```
shape.MoveTo(50,50)
Clr = LXColor.new(255,0,0,255)
shape.LineTo(100,100,Clr)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

DrawEllipse

绘制圆\椭圆

```
DrawEllipse(rect, clr, dWidth, dPattern, dPatternScale)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	圆\椭圆的矩形边界	否	无
clr	LXColor	颜色值	否	无
dWidth	number	线宽	是	1
dPattern	number	线样式	是	-1
dPatternScale	number	线缩放因子	是	1

返回值

类型	描述
number	绘制是否成功, 0 失败, 1 成功

注意事项

dWidth 默认值为 1
dPattern 默认值为-1
dPatternScale 默认值为 1

示例

```
shape = LX2DShape.new()
Clr = LXColor.new(255,0,0,255)
Rect = LXRect.new(100,100,500,300)            --绘制一个椭圆
shape.DrawEllipse(Rect,clr)
Rect1 = LXRect.new(0,0,100,100)
shape.DrawEllipse(Rect,Clr)                   --绘制一个圆
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

FillEllipse

填充椭圆

```
FillEllipse(rect, clr)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	椭圆的矩形边界	否	无
clr	LXColor	颜色值	否	无

返回值

无

注意事项

无

示例

```
shape = LX2DShpae.new()
clr = LXColor.new(255,0,0,255)
```

```
Rect = LXRect.new(100,100,500,300)
shape.FillEllipse(Rect,clr)                    --绘制一个填充色椭圆
Rect1 = LXRect.new(0,0,100,100)
shape.DrawEllipse(Rect,clr)                    --绘制一个填充色圆
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua
```

DrawChord

绘制弦

```
DrawChord(rect, PtStart, PtEnd, clr, dWidth, dPattern, dPatternScale)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	弦的矩形边界	否	无
PtStart	LXVector2	弦的开始点	否	无
PtEnd	LXVector2	弦的结束点	否	无
clr	LXColor	颜色值	否	无
dWidth	number	线宽	是	1
dPattern	number	线样式	是	-1
dPatternScale	number	线缩放因子	是	1

返回值

类型	描述
boolean	是否绘制成功

注意事项

```
无
```

示例

```
shape = LX2DShape.new()
PtStart = LXVector2.new(50,20)
PtEnd =    LXVector2.new(300,400)
Rect = LXRect.new(100,100,400,500)
Clr = LXColor.new(255,0,10,255)
shape.DrawChrod(Rect, PtStart, PtEnd, Clr)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

FillChord

填充弦

FillChord(rect, PtStart, PtEnd, clr)

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	弦的矩形边界	否	无
PtStart	LXVector2	弦的开始点	否	无
PtEnd	LXVector2	弦的结束点	否	无
clr	LXColor	颜色值	否	无

返回值

无

注意事项

无

示例

```
shape = LX2DShape.new()
PtStart = LXVector2.new(50,20)
PtEnd =    LXVector2.new(300,400)
Rect = LXRect.new(100,100,400,500)
Clr = LXColor.new(255,0,10,255)
shape.FillChord(Rect, PtStart, PtEnd, Clr)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

DrawRoundRect

绘制圆角矩形

DrawRoundRect(rect, Pt, clr, dWidth, dPattern, dPatternScale)

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	矩形	否	无

Pt	LXVector2	确定椭圆四个圆角的形状	否	无
clr	LXColor	颜色值	否	无
dWidth	number	线宽	是	1
dPattern	number	线样式	是	-1
dPatternScale	number	线缩放因子	是	1

返回值

类型	描述
boolean	是否绘制成功

注意事项

dWidth 默认值为 1
dPattern 默认值为-1
dPatternScale 默认值为 1

示例

```
shape = LX2DShape.new()
Pt = LXVector2.new(50,20)
Rect = LXRect.new(100,100,500,500)
clr = LXColor.new(255, 0, 0, 255)
shape.DrawRoundRect(Rect, Pt, clr)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

FillRoundRect

填充圆角矩形

```
FillRoundRect(rect, Pt, clr)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	矩形	否	无
Pt	LXVector2	确定圆角的大小	否	无
Clr	LXColor	颜色值	否	无

返回值

类型	描述
number	绘制是否成功, 0 失败, 1 成功

注意事项

无

示例

```
shape = LX2DShape.new()
Pt = LXVector2.new(50,20)
Rect = LXRect.new(100,100,500,500)
clr = LXColor.new(255, 0, 0, 255)
shape.FillRoundRect(Rect, Pt, clr)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

DrawArc

绘制弧

```
DrawArc(rect, PtStart, PtEnd, clr， dWidth, dPattern, dPatternScale)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	弦的矩形边界	否	无
PtStart	LXVector2	弦的开始点	否	无
PtEnd	LXVector2	弦的结束点	否	无
clr	LXColor	颜色值	否	无
dWidth	number	线宽	是	1
dPattern	number	线样式	是	-1
dPatternScale	number	线缩放因子	是	1

返回值

无

注意事项

无

示例

```
shape = LX2DShape.new()
PtStart = LXVector2.new(50,20)
PtEnd =    LXVector2.new(300,400)
Rect = LXRect.new(100,100,400,500)
Clr = LXColor.new(255,0,10,255)
shape.DrawArc(Rect, PtStart, PtEnd, Clr)
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua
```

SetViewPortBKColor

设置当前视区的背景颜色

```
SetViewPortBKColor(color)
SetViewPortBKColor(r, g, b, a)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色对象	否	无
r	number	颜色 r 分量	否	无
g	number	颜色 g 分量	否	无
b	number	颜色 b 分量	否	无
a	number	颜色 a 分量	是	255

返回值

```
无
```

注意事项

```
无
```

示例

```
shape = LX2DShape.new()
shape:SetViewPortBKColor(LXColor.new(255, 0, 0, 255))
shape:SetViewPortBKColor(255, 0, 0)
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua
```

GetViewPortBKColor

获取当前视区的背景颜色

GetViewPortBKColor()

参数

无

返回值

类型	描述
LXColor	颜色对象

注意事项

无

示例

```
shape = LX2DShape.new()
bkcolor = shape.GetViewPortBKColor()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LX2DShape\脚本\XMain.lua

LXTexture

功能描述

该对象实现了 2 维图像的创建，绘制，保存和基本的图像处理。

函数列表

函数名称	说明
new	创建一个 LXTexture 对象
Blt	绘制纹理
StretchBlt	拉伸绘制纹理
StretchBlt	利用一张掩码图绘制纹理
RenderBlt	高级绘制纹理
RenderTranslate	应用变换矩阵绘制纹理
DivdBlt	根据给定区域和索引裁减图像后绘制
StretchBlendBlt	混合绘制纹理
RenderBlendBlt	高级混合绘制纹理
Copy	拷贝源图像指定帧指定区域到当前图像指定帧指定区域
SetColor	设置图像像素点的颜色值
GetFrameCount	获取图像的桢数
GetFrame	拷贝指定纹理指定帧到源纹理指定帧
StepToFrame	切换到指定桢
GetCurrentFrame	得到当前显示的帧索引
InsertFrame	向动画中插入某一桢
ReplaceFrame	替换动画中的某一桢
DeleteFrame	删除动画中的一桢
CopyFromScreen	拷贝屏幕图像
SetLoop	设置是否循环标志
ReSet	重置纹理
GetBeEndPlay	得到是否结束标志
GetTotalDelay	得到当前动画的总延时
GetWidth	获取图像的宽度
GetHeight	获取图像的高度
GetFormat	获取图像的像素格式
GetColor	获取图像像素点的颜色值
SaveToFile	保存图像到指定目录下
Play	播放纹理
Stop	停止播放纹理
GetImage	得到指定帧对应的内存位图

new

创建一个 LXTexture 对象

```
new(Handle)
new(FileName)
new(Width, Height, Format, Name)
```

参数

参数名	类型	描述	是否可选	默认值
Handle	LXhandle	图像句柄	否	无

参数名	类型	描述	是否可选	默认值
FileName	string	图像文件路径	否	无

参数名	类型	描述	是否可选	默认值
Width	number	图像宽度	否	无
Height	number	图像高度	否	无
Format	string	图像像素格式	否	无
Name	string	图像名称	是	系统生成

返回值

无

注意事项

Format 取值以下几种：
"R5G6B5FORMAT"　　表示 565 16 位色格式
"A1R5G5B5FORMAT"　　表示 555 带位通道的 16 位色格式
"A4R4G4B4FORMAT"　　表示 4444 16 位色格式
"A8R8G8B8FORMAT"　　表示带通道的 32 位色格式

示例

```
--第一种创建方式
local handle = LXHandle.new(“..\\资源\\1.jpg”)
local Texture = LXTexture.new(handle)

--第二种创建方式
local Texture = LXTexture.new(“..\\资源\\1.jpg”)

--第三种创建方式
local Texture = LXTexture.new(128, 128, “R5G6B7FORMAT”)
```

参见

Samples\Test_Lualib\Test_LXTexture\Test_LXTexture.lua

Blt

绘制纹理

```
Blt(DestLeft,DestTop,lxSrcRect,OutputFlags,Rotate,MirrorType, BeLiner, ColorDiffuse)
```

参数

参数名	类型	描述	是否可选	默认值
DestLeft	number	绘制区域左坐标	否	无
DestTop	number	绘制区域上坐标	否	无
lxSrcRect	LXRect	源图像矩形区域	是	nil
OutputFlags	string	应用纹理 Alpha 显示	是	nil
Rotate	number	旋转的角度	是	0
MirrorType	number	镜像类型	是	0
BeLiner	Boolean	是否线性采样	是	true
ColorDiffuse	LXColor	偏置颜色	是	白色

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

OutputFlags 取值如下:
'a'表示运用 Alpha 显示
'r'表示反向 Alpha 显示
'A'表示输出 Alpha +
's'表示输出 Alpha －
'm'表示输出 Alpha *
'M' 表示 Multiple /

MirrorType 取值如下
=0 不镜像
=1 X 镜像
=2 X 镜像
=3 XY 镜像

示例

```
rcSrc = LXRect.new(0, 0, 500, 500)
BeSucc = texture.Blt(0, 0, rcSrc, "a", 0, 0, false, color)
--顺时针旋转 90 度绘制
BeSucc = texture.Blt(0, 0, rcSrc, "a", 90)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

StretchBlt

拉伸绘制纹理

StretchBlt (lxDesRect, lxSrcRect, OutputFlags, Rotate ， MirrorType, BeLiner, ColorDiffuse)

参数

参数名	类型	描述	是否可选	默认值
lxDesRect	LXRect	绘制到屏幕的矩形区域	是	nil
lxSrcRect	LXRect	源图像的矩形区域	是	nil
OutputFlags	string	是否应用文理 Alpha 显示	是	nil
Rotate	number	旋转的角度	是	0
MirrorType	number	镜像类型	是	0
BeLiner	Boolean	是否线性采样	是	true
ColorDiffuse	LXColor	偏置颜色	是	白色

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

OutputFlags, Rotate， MirrorType, BeLiner，ColorDiffuse 参见函数 Blt 中的说明

示例

```
rcDes = LXRect.new(0, 0, 500, 500)
texture.StretchBlt(rcDes)

RcSrc = LXRect.new(0, 0, 500, 500)
Texture. StretchBlt(rcDes, rcSrc)

BeSucc = texture.StretchBlt(rcDes, nil, nil, "a", 0, 0, true)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

StretchBltMask

绘制带掩码图纹理

```
StretchBltMask (lxMaskTexture, DesRect, SrcRect, MaskRect, Rotate, MirrorType)
```

参数

参数名	类型	描述	是否可选	默认值
lxMaskTexture	LXHandle	掩码图句柄	否	无
DesRect	LXRect	绘制矩形区域	是	nil
SrcRect	LXRect	源图像矩形区域	是	nil
MaskRect	LXRect	掩码图矩形区域	是	nil
Rotate	number	旋转的角度	是	0
MirrorType	number	镜像类型	是	0

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

Rotate， MirrorType 参见函数 Blt 中的说明

示例

```
BeSucc = texture.StretchBltMask(Handle, rcDes, rcSrc,nil,90,0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

RenderBlt

高级绘制纹理

```
RenderBlt (lxSrcRect, lxDesRect, ColorDiffuse, DrawFlags,OutputFlags, RotateAngle, Depth, MirrorType)
```

参数

参数名	类型	描述	是否可选	默认值
lxSrcRect	LXRect	源图像的矩形区域	是	nil
lxDesRect	LXRect	绘制到屏幕的矩形区域	是	nil

ColorDiffuse	LXColor	偏置颜色	是	白色
DrawFlags	string	渲染纹理混合标志	是	nil
OutputFlags	string	应用纹理 Alpha 显示	是	nil
RotateAngle	number	旋转的角度	是	0
Depth	number	按照绘制顺序渲染	是	0
MirrorType	number	镜像类型	是	0

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

ColorDiffuse, RotateAngle, MirrorType 参考 Blt 函数参数说明

DrawFlags 为渲染纹理混合标志 由单个字符组成 可组合使用
'd':Color 第一个参数使用颜色
't':Color 第一个参数使用纹理
'c':Color 第一个参数使用当前值
'p':Alpha 第一个参数使用当前值
'h':Alpha 第一个参数使用纹理
's':Alpha 第一个参数使用当前值
'D':Color 第二个参数使用颜色
'T':Color 第二个参数使用纹理
'C':Color 第二个参数使用当前值
'P':Alpha 第二个参数使用颜色
'H':Alpha 第二个参数使用纹理
'S':Alpha 第二个参数使用当前值
'r':传送第一个参数颜色
'e':传送第二个参数颜色
'n':禁止输出颜色
'm':输出颜色乘
'a':输出颜色+
'u':输出颜色-
'l':输出颜色 Alpha 混合(使用颜色中的 Alpha 值)
'z':输出颜色 Alpha 混合(使用纹理中的 Alpha 值)
'x':输出颜色 Alpha 混合(使用当前状态中的 Alpha 值)
'w':输出颜色乘加亮倍
'f':输出颜色乘加亮倍
'o':输出颜色平滑+
'y':传送第一个参数 Alpha
'Y':传送第二个参数 Alpha

'F':禁止输出 Alpha
'M':输出 Alpha 乘
'A':输出 Alpha+
'U':输出 Alpha-

OutputFlags 为渲染纹理输出标志　由单个字符组成　可组合使用
'e':输出使能 AlphaTest
'E':输出如果 AlphaTest 则源 Alpha 值>1　通过
's':输出如果 AlphaTest 则源 Alpha 值<254　通过
'a':输出 Alpha 效果
'A':混合+
'S':混合-
'r':混合反向-
'm':混合取最小值
'M':混合取最大值
'z':混合源使用
'o':混合源使用
'p':混合源使用源 Alpha
'R':混合源使用反向源 Alpha
't':混合源使用目标 Alpha
'T':混合源使用反向目标 Alpha
'B':混合源使用源颜色
'c':混合源使用反向源颜色
'n':混合源使用目标颜色
'N':混合源使用反向目标颜色
'Z':混合目标使用
'O':混合目标使用
'h':混合目标使用源 Alpha
'H':混合目标使用反向源 Alpha
'P':混合目标使用目标 Alpha
'w':混合目标使用反向目标 Alpha
'x':混合目标使用源颜色
'X':混合目标使用反向源颜色
'Y':混合目标使用目标颜色
'v':混合目标使用反向目标颜色
'b':支持抖动
'L':应用线性采样

示例

```
rcSrc = LXRect.new(0, 0, 256, 256)
rcDes = LXRect.new(0, 0, 200, 200)
color = LXColor.new(255, 255, 0, 0)
texture.RenderBlt(rcSrc, rcDes, color, "dpTHa", "Eb", 1, 0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

RenderTranslate

高级绘制纹理　应用变换矩阵

RenderTranslate (lxFirVector, lvSecVector, lxSrcRect, ColorDiffuse, DrawFlags, OutputFlags, Depth， MirrorType, InverseCull)

参数

参数名	类型	描述	是否可选	默认值
lxFirVector	LXVector3	变换矩阵列向量	是	(1,0,0)
lxSecVector	LXVector3	变换矩阵列向量	是	(0,1,0)
lxSrcRect	LXRect	源图像的矩形区域	是	nil
ColorDiffuse	LXColor	偏置颜色	是	白色
DrawFlags	string	渲染纹理混合标志	是	nil
OutputFlags	string	是否应用文理 Alpha 显示	是	nil
Depth	number	按照绘制顺序渲染	是	0
MirrorType	number	镜像类型	是	0
InverseCull	Boolean	是否逆向裁剪	是	false

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

无

示例

```
rcSrc = LXRect.new(0, 0, 256, 256)
rcDes = LXRect.new(0, 0, 200, 200)
Vec1 = LXVector3.new(1, 1, 2)
Vec2 = LXVector3.new(2, 3, 4)
texture.RenderTranslate(Vec1, Vec2, rcSrc, color, "", "", 0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

DivdBlt

根据给定区域和索引裁减图像后绘制

DivdBlt (DivdWidth, DivdHeight)

DivdBlt (DivdWidth, DivdHeight, SrcIndex, DesLeft, DesTop, OutputFlags, Rotate, MirrorType, BeLiner)

DivdBlt (DivdWidth, DivdHeight, SrcIndex, lxRect, OutputFlags, Rotate, MirrorType, BeLiner)

参数

参数名	类型	描述	是否可选	默认值
DivdWidth	number	源图像裁剪成小图像的宽度	否	无
DiveHeight	number	源图像裁剪成小图像的高度	否	无

参数名	类型	描述	是否可选	默认值
DivdWidth	number	源图像裁剪成小图像的宽度	否	无
DiveHeight	number	源图像裁剪成小图像的高度	否	无
SrcIndex	number	需要绘制小图像列表中图像索引	是	0
DesLeft	number	绘制到屏幕 X 坐标	是	0
DesTop	number	绘制到屏幕 Y 坐标	是	0
OutputFlags	string	是否应用 Alpha 显示	是	nil
Rotate	number	旋转角度	是	0
MirrorType	number	镜像类型	是	0
BeLiner	Boolean	是否线性采样	是	true

参数名	类型	描述	是否可选	默认值
DivdWidth	number	源图像裁剪成小图像的宽度	否	无
DiveHeight	number	源图像裁剪成小图像的高度	否	无
SrcIndex	number	需要绘制小图像列表中图像索引	是	0
lxRect	LXRect	绘制到屏幕矩形区域	是	nil
OutputFlags	string	是否应用 Alpha 显示	是	nil

Rotate	number	旋转角度	是	0
MirrorType	number	镜像类型	是	0
BeLiner	Boolean	是否线性采样	是	true

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

OutputFlags, Rotate, MirrorType, BeLiner 参见 Blt 函数中的说明

示例

```
BeSucc = texture.DivdBlt(42, 38, 9, rc, "a", 0, 1)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

StretchBlendBlt

混合绘制纹理

```
StretchBlendBlt (xBlendHandle, lxDesRect, lxSrcRect, OutputFlags, Rotate, MirrorType, BeLiner, ColorDiffuse)
```

参数

参数名	类型	描述	是否可选	默认值
xBlendHandle	LXHandle	混合图像句柄	否	无
lxDesRect	LXRect	绘制到屏幕矩形区域	是	nil
lxSrcRect	LXRect	源图像的矩形区域	是	nil
OutputFlags	string	是否应用 Alpha 显示	是	nil
Rotate	number	旋转角度	是	0
MirrorType	number	镜像类型	是	0
BeLiner	Boolean	是否线性采样	是	true
ColorDiffuse	LXColor	偏置颜色	是	白色

返回值

类型	描述

number	绘制成功返回 0，失败返回非 0

注意事项

无

示例

```
Handle = LXHandle.new()
Handle.Load("Terrain0.xmp")
rcSrc = LXRect.new(0, 0, 500, 500)
rcDes = LXRect.new(0, 0, 500, 500)
BeSucc = texture.StretchBlt(rcDes, nil, nil, "a", 0, 0, true)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

RenderBlendBlt

高级混合绘制纹理

```
RenderBlendBlt (xBlendHandle, lxSrcRect, lxBlendRect, lxDesRect, ColorDiffuse, SrcFlags, DesFlags, OutputFlags, RotateAngle, Depth, MirrorType)
```

参数

参数名	类型	描述	是否可选	默认值
xBlendHandle	LXHandle	混合图像句柄	否	无
lxSrcRect	LXRect	源图像的矩形区域	是	nil
lxBlendRect	LXRect	混合图像矩形区域	是	nil
lxDesRect	LXRect	绘制到屏幕矩形区域	是	nil
ColorDiffuse	LXColor	偏置颜色	是	白色
SrcFlags	string	源图像渲染标志	是	nil
DesFlags	string	目标纹理渲染标志	是	nil
OutputFlags	string	输出标志	是	nil
RotateAngle	number	旋转角度	是	0
Depth	number	是否按照顺序绘制	是	0
MirrorType	number	镜像类型	是	0

返回值

类型	描述

number	绘制成功返回 0，失败返回非 0

注意事项

SrcFlags DesFlags OutputFlags　MirrorType Depth 的参数说明参见 RenderBlt 函数

示例

```
if Handle == nil then
	Handle = LXHandle.new()		Handle.Load("lbase_boots.bmp")
End
rcSrc = LXRect.new(0, 0, 256, 256)
rcDes = LXRect.new(0, 0, 200, 200)
rcBlend = LXRect.new(0, 0, 256, 256)
color = LXColor.new(255, 255, 0, 0)
BeSucc = texture.RenderBlendBlt(Handle, rcSrc, rcBlend, rcDes, color, "", "", "", 0, 0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

Copy

拷贝源图像指定帧指定区域到当前图像指定帧指定区域

```
Copy (Handle, SrcIndex, DesIndex, DesRect, SrcRec)
```

参数

参数名	类型	描述	是否可选	默认值
Handle	LXHandle	源图像句柄	否	无
SrcIndex	number	源图像指定帧索引	是	0
DesIndex	number	当前图像指定帧索引	是	0
DesRect	LXRect	当前图像指定绘制区域	是	nil
SrcRect	LXRect	源图像选择区域	是	nil

返回值

类型	描述
number	拷贝成功返回 0，失败返回非 0

注意事项

无

示例

```
texture = LXTexture.new()
texture.CreateFromFile("l_base_body.bmp")

texture1 = LXTexture.new()
texture1.Create(256, 256)
texture1.Copy(texture, 0, 0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

SetColor

设置图像像素点的颜色值

```
SetColor (PixX, PixY, Color, FrmIndex)
```

参数

参数名	类型	描述	是否可选	默认值
PixX	number	像素点 X 坐标	否	无
PixY	number	像素点 Y 坐标	否	无
Color	LXColor	指定颜色值	是	白色
FrmIndex	number	指定帧	是	0

返回值

类型	描述
number	设置成功返回 0，失败返回非 0

注意事项

无

示例

```
color = LXColor.new(255, 255, 0, 0)
```

```
texture.SetColor(10, 10, color)
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua
```

GetFrameCount

获取图像的桢数

```
GetFrameCount ()
```

参数

```
无
```

返回值

类型	描述
number	总帧数

注意事项

```
无
```

示例

```
Count = texture.GetFrameCount()
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua
```

GetFrame

拷贝源纹理指定帧到指定纹理指定帧

```
GetFrame (Index, Handle, SrcIndex)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	当前纹理指定帧索引	否	无
Handle	number	源图像	否	无
SrcIndex	number	源图像指定帧索引	是	0

返回值

类型	描述
number	获取成功返回 0，失败返回非 0

注意事项

无

示例

```
texture.GetFrame(0, texture1, 0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

StepToFrame

切换到指定桢

```
StepToFrame (Index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	当前纹理指定帧索引	否	无

返回值

类型	描述
number	切换成功返回 0，失败返回非 0

注意事项

无

示例

```
texture.StepToFrame(0)
```

参见

Samples\Test_Lualib\Test_LXTexture\Test_LXTexture.lua

GetCurrentFrame

得到当前显示的帧索引

```
GetCurrentFrame ()
```

参数

无

返回值

类型	描述
number	当前显示的帧索引

注意事项

无

示例

```
local index = texture.GetCurrentFrame()
print(index)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

InsertFrame

向动画中插入某一桢

```
InsertFrame (Index, Handle, SrcInde)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	当前纹理指定帧索引	否	无
Handle	number	插入图像	否	无
SrcIndex	number	插入图像指定帧索引	否	无

返回值

类型	描述

number	插入成功返回 0，失败返回非 0

注意事项

无

示例

```
--将 texture1 的第一帧图像插入 texture 成为其第二帧图像
texture.InsertFrame(0, texture1, 0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

ReplaceFrame

替换动画中的某一桢

```
ReplaceFrame (Index, Handle, SrcIndex)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	当前纹理指定帧索引	否	无
Handle	number	替换图像	否	无
SrcIndex	number	替换图像指定帧索引	是	0

返回值

类型	描述
number	替换成功返回 0，失败返回非 0

注意事项

无

示例

```
--将 texture 的第一帧图像替换成 texture1 的第一帧图像
Res = texture.ReplaceFrame(0, texture1, 0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

DeleteFrame

删除动画中的一桢

```
DeleteFrame (Index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	当前纹理指定帧索引	否	无

返回值

类型	描述
number	替换成功返回 0，失败返回非 0

注意事项

无

示例

```
Res = texture.DeleteFrame(0)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

CopyFromScreen

拷贝屏幕图像

```
CopyFromScreen ()
```

参数

无

返回值

类型	描述

number	拷贝成功返回 0，失败返回非 0

注意事项

无

示例

```
Res = texture.CopyFromScreen()
if Res == 0 then
texture.SaveToFile("a.xmp")
end
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

SetLoop

设置是否循环标志

```
SetLoop (bLoop)
```

参数

参数名	类型	描述	是否可选	默认值
bLoop	Boolean	是否循环播放	否	无

返回值

无

注意事项

无

示例

```
Texture1.SetLoop(true)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

ReSet

重置

```
ReSet()
```

参数

无

返回值

无

注意事项

无

示例

```
Texture1.Reset()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

GetBeEndPlay

得到是否结束标志

```
GetBeEndPlay ()
```

参数

无

返回值

类型	描述
Boolean	是否播放结束

注意事项

无

示例

```
bEnd = Texture1.GetBeEndPlay()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

GetTotalDelay

得到当前动画的总延时

GetTotalDelay ()

参数

无

返回值

类型	描述
number	动画总延时

注意事项

无

示例

Delay = Texture1.GetTotalDelay()

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

GetWidth

获取图像的宽度

GetWidth ()

参数

无

返回值

类型	描述
number	图像的宽度

注意事项

无

示例

```
Width = texture.GetWidth()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

GetHeight

获取图像的高度

```
GetHeight ()
```

参数

无

返回值

类型	描述
number	图像的高度

注意事项

无

示例

```
Height = texture.GetHeight()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

GetFormat

获取图像的像素格式

```
GetFormat ()
```

参数

无

返回值

类型	描述
number	像素格式

注意事项

Format 取值如下:
= 0　　未知格式
= 1　　565 16 位色格式
= 2　　555 带位通道的 16 位色格式
= 3　　4444 16 位色格式

示例

```
Format = texture.GetFormat()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

GetColor

获取图像像素点的颜色值

```
GetColor (PixX, PixY, FrmIndex)
```

参数

参数名	类型	描述	是否可选	默认值
PixX	number	像素点 X 坐标	否	无
PixY	number	像素点 Y 坐标	否	无
FrmIndex	number	指定帧	是	0

返回值

类型	描述
LXColor	颜色值

注意事项

无

示例

```
Color = texture.GetColor(10, 10)
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua
```

SaveToFile

保存图像到指定目录下

```
SaveToFile (FileName)
```

参数

参数名	类型	描述	是否可选	默认值
FileName	string	文件路径	否	无

返回值

类型	描述
number	保存成功返回 0，失败返回非 0

注意事项

```
无
```

示例

```
texture.SaveToFile("D:\\a.xmp")
```

参见

```
Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua
```

Play

播放纹理

```
Play ()
```

参数

```
无
```

返回值

```
无
```

注意事项

无

示例

```
texture.Play()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

Stop

停止播放纹理

```
Stop()
```

参数

无

返回值

无

注意事项

无

示例

```
texture.Stop()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

GetImage

得到指定帧对应的内存位图

```
GetImage(index)
```

参数

参数名	类型	描述	是否可选	默认值
index	number	帧索引	否	无

返回值

类型	描述
LXImage	内存位图 LXImage 对象

注意事项

无

示例

```
local img = texture2.GetImage(0)
print(img)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXTexture\脚本\XMain.lua

LXImage

功能描述

该对象可以在内存中创建内存位图对象，并对其内容进行编辑和保存操作，用户可以将其做为纹理输出到屏幕前的一个缓冲

函数列表

函数名称	说明
new	创建一个 LXImage 对象
GetWidth	得到内存位图的宽度
GetHeight	得到内存位图的高度
SaveToFile	将内存位图指定区域保存为指定格式的文件
Blt	将源内存位图指定区域的内容拷贝到当前位图指定区域
BltWithRgn	带裁剪区的拷贝函数
ColorFill	填充当前位图指定区域的颜色
Destroy	删除创建的内存位图对象

new

创建一个 LXImage 对象

```
new(FileName)
new(Width, Height, Pool, Format)
```

参数

参数名	类型	描述	是否可选	默认值
FileName	string	图像文件名	否	无

参数名	类型	描述	是否可选	默认值
Width	number	内存位图的宽度	否	无
Height	number	内存位图的高度	否	无
Pool	number	位图所在内存位置	是	1
Format	string	内存位图图像格式	是	4

返回值

类型	描述
LXImage	创建出来的 LXImage 对象

注意事项

Pool 取值如下:
= 1　　系统内存
= 2　　管理内存
= 3　　显示内存

Formatq 取值如下:
= 1　　R5G6B5 格式
= 2　　A1R5G5B5 格式
= 3　　A4R4G4B4 格式
= 4　　A8R8G8B8 格式

从一个文件创建内存位图对象时，该图像文件的像素格式如果为 DTX 压缩格式，则创建不能成功。

示例

```
img = LXImage.new(“..\\资源\\黑车.bmp”)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

GetWidth

得到内存位图的宽度

```
GetWidth()
```

参数

无

返回值

类型	描述
number	内存位图的宽度

注意事项

无

示例

```
img = LXImage.new(“..\\资源\\黑车.bmp”)
width = img.GetWidth()
print(width)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

GetHeight

得到内存位图的高度

```
GetHeight()
```

参数

无

返回值

类型	描述
number	内存位图的高度

注意事项

无

示例

```
img = LXImage.new(“..\\资源\\黑车.bmp”)
height = img.GetHeight ()
print(height)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

SaveToFile

将内存位图指定区域保存为指定格式的文件

```
SaveToFile(FileName, Format, Rect)
```

参数

参数名	类型	描述	是否可选	默认值
FileName	string	文件名	否	无
Format	string	图像格式	是	“jpg”
Rect	LXRect	指定区域	是	nil

返回值

类型	描述
boolean	是否保存成功

注意事项

Format 取值如下:
= “jpg”　　jpg 格式
= “xmp”　　xmp 格式
= “bmp”　　bmp 格式
= “png”　　png 格式

示例

```
img = LXImage.new( “..\\资源\\黑车.bmp” )
--将 img 保存为 1.jpg 文件
img.SaveToFile(“D:\\1.jpg”)
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

Blt

将源内存位图指定区域内容拷贝到当前位图指定区域

```
Blt(DesRect, SrcImage, SrcRect, bDither, FilterType, MirrorType, ColorKey)
```

参数

参数名	类型	描述	是否可选	默认值
DesRect	LXRect	当前位图指定区域	否	无
SrcImage	LXImage	源内存位图	否	无
SrcRect	LXRect	源内存位图指定区域	否	无
bDither	boolean	是否抖动处理	是	true
FilterType	number	过滤方式	是	2
MirrorType	number	镜像方式	是	0
ColorKey	LXColor	颜色键	是	黑色

返回值

类型	描述
boolean	是否拷贝成功

注意事项

FilterType 取值如下:
= 1　　最近点过滤
= 2　　线性过滤
= 3　　各向异性过滤
MirrorType 取值如下:
= 0　　不使用镜像
= 1　　水平镜像
= 2　　垂直镜像
颜色键指定的值会被置为黑色

示例

```
img = LXImage.new(“..\\资源\\黑车.bmp”)
BkImage = LXImage.new(64,64)
BkImage.Blt(LXRect.new(0,0,20,20), img, LXRect(0,0,64,64))
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

BltWithRgn

带裁剪区的拷贝函数

```
Blt(OffsetX, OffsetY, DestRgn, SrcImage, SrcRect, bApplySrcAlpha, bApplyRotate, fRotate, bDither, bApplySrcClrKey, ColorKey)
```

参数

参数名	类型	描述	是否可选	默认值
OffsetX	number	目标区域左上角 X 坐标	否	无
OffsetY	number	目标区域左上角 Y 坐标	否	无
DestRgn	LXRgn	当前位图指定区域	否	无
SrcImage	LXImage	源内存位图	否	无
SrcRect	LXRect	源内存位图指定区域	是	nil
bApplySrcAlpha	boolean	是否应用源图像 Alpha	是	false
bApplyRotate	boolean	是否应用旋转	是	false
fRotate	number	旋转的角度（弧度）	是	0
bDither	boolean	是否抖动处理	是	false
bApplySrcClrKey	boolean	是否应用源图像颜色键	是	false
ColorKey	LXColor	颜色键	是	黑色

返回值

类型	描述
boolean	是否拷贝成功

注意事项

FilterType 取值如下:
= 1　　最近点过滤
= 2　　线性过滤
= 3　　各向异性过滤
MirrorType 取值如下:
= 0　　不使用镜像
= 1　　水平镜像
= 2　　垂直镜像
颜色键指定的值会被置为黑色

示例

```
img = LXImage.new(“..\\资源\\黑车.bmp”)
BkImage = LXImage.new(64,64)
BkImage.Blt(LXRect.new(0,0,20,20), img, LXRect(0,0,64,64))
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

ColorFill

填充当前位图指定区域的颜色

```
ColorFill(Rect, Color)
```

参数

参数名	类型	描述	是否可选	默认值
Rect	LXRect	指定区域	否	无
Color	LXColor	指定颜色	否	无

返回值

类型	描述
boolean	填充是否成功

注意事项

无

示例

```
img = LXImage.new(“..\\资源\\黑车.bmp”)
BkImage = LXImage.new(64,64)
BkImage.Blt(LXRect.new(0,0,20,20), img, LXRect(0,0,64,64))
BkImage.ColorFill(LXRect.new(20,20,40,40), LXColor.new(255,0,0))
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

Destroy

删除创建的内存位图对象

```
Destroy()
```

参数

无

返回值

无

注意事项

无

示例

```
img = LXImage.new(“..\\资源\\黑车.bmp”)
img.Destroy()
```

参见

Samples\Test_LuaLib\Test_2DGraph\Test_LXImage\脚本\XMain.lua

LXSound

功能描述

该对象提供声音文件的播放及控制功能

函数列表

函数名称	说明
new	创建一个 LXSound 对象
Play	播放
Destroy	删除声音对象
Reset	重新开始播放
Stop	停止

new

创建一个 LXSound 对象

```
new(SoundName)
```

参数

参数名	类型	描述	是否可选	默认值
SoundName	string	声音文件名	否	无

返回值

类型	描述
LXSound	创建出来的 LXSound 对象

注意事项

无

示例

```
sound = LXSound.new(“D:\\1.mp3”)
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXSound\脚本\XMain.lua

Play

播放声音

```
Play(BeLoop, BeDeleteOnEnd)
```

参数

参数名	类型	描述	是否可选	默认值
BeLoop	Boolean	是否循环播放	否	无
BeDeleteOnEnd	Boolean	是否在播放完后删除对象	是	true

返回值

类型	描述
Boolean	是否播放成功

注意事项

无

示例

```
bRes = sound.Play(true)
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXSound\脚本\XMain.lua

Destroy

删除声音对象

```
Destroy()
```

参数

无

返回值

类型	描述
Boolean	是否删除成功

注意事项

无

示例

```
bRes = sound.Destroy()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXSound\脚本\XMain.lua a

Reset

重新开始播放

```
Reset()
```

参数

无

返回值

类型	描述
Boolean	重新播放是否成功

注意事项

无

示例

```
bRes = sound.Reset()
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXSound\脚本\XMain.lua
```

Stop

停止播放

```
Stop()
```

参数

```
无
```

返回值

类型	描述
boolean	停止播放是否成功

注意事项

```
无
```

示例

```
bRes = sound.Stop()
```

参见

```
Samples\Test_LuaLib\Test_XBase\Test_LXSound\脚本\XMain.lua
```

BeEnd

是否播放完毕

```
BeEnd()
```

参数

```
无
```

返回值

类型	描述

boolean	是否播放完毕

注意事项

无

示例

```
bRes = sound.BeEnd()
```

参见

Samples\Test_LuaLib\Test_XBase\Test_LXSound\脚本\XMain.lua

3.4.3 网络功能

网络开发简述

X 魔方应用中的网络开发比较简洁。

网络功能目前由导出类(LXNetServer 和 LXNetClient)和网络事件组成。
网络事件请参考 Lua 事件部分。

下面介绍一下其基本流程。

X 魔方应用网络应用分为服务器端与客户端。

服务器端，首先创建一个 LXNetServer 对象，然后调用 Start(nPort)函数启动对某个端口的监听，如果停止服务调用 Stop。通过 X_ONNEWCONNECT 事件响应新的连接；通过 X_ONRECEICEMSG 响应接受到的消息；通过 X_ONCONNECTBREAK 事件响应连接断开。在上述三个事件中，Param 参数关联或对应着一个 LXNetClient 对象，它表示对应着某个客户端的一个连接，服务器端可以通过此对象向对应的客户端发送数据。

客户端，首先创建一个 LXNetClient 对象，然后调用 Connect(serverip, port)函数建立起客户端与服务器的连接。通过 Send()函数发送消息。同样地，通过上述事件来响应接受消息，断开消息等。

网络开发的简单示例可参考：
Samples\Test_LuaBase\Test_LXNetServer
Samples\Test_LuaBase\Test_LXNetClient
网络开发的应用示例可参考：Samples/Games/2DGames/Chess 中国象棋网络对战

LXNetServer

功能描述

实现网络中的服务器端。

函数列表

函数名称	说明
new	创建对象
Start	开始服务
Stop	停止服务

new

创建一个 LXNetServer 对象

```
new()
```

参数

无

返回值

类型	描述
LXNetServer	LXNetServer 对象

注意事项

无

示例

```
server = LXNetServer.new()
```

参见

Samples\Test_LuaBase\Test_LXNetServer\脚本\XMain.lua

Start

启动服务

```
Start(nPort)
```

参数

参数名	类型	描述	是否可选	默认值
nPort	number	服务器监听端口	否	无

返回值

类型	描述
number	错误号： =0 成功 =1 内部创建线程或事件错 =2 超时错 =3 外部关闭了 Socket =4 连接错 =5 参数错 =6 上一次连接未关闭 =7 监听错

注意事项

无

示例

```
local nErr = server.Start(2005)
if 0 ~= nErr then
	XMessageBox("Sever Start Erro")
else
	print("Server now started")
end
```

参见

Samples\Test_LuaBase\Test_LXNetServer\脚本\XMain.lua

Stop

停止服务

```
Stop()
```

参数

无

返回值

类型	描述
number	错误号： =0 成功 =1 内部创建线程或事件错 =2 超时错 =3 外部关闭了 Socket =4 连接错 =5 参数错 =6 上一次连接未关闭 =7 监听错

注意事项

无

示例

```
local nErr = server.Stop()
```

参见

Samples\Test_LuaBase\Test_LXNetServer\脚本\XMain.lua

LXNetClient

功能描述

实现网络中的客户端。

函数列表

函数名称	说明
new	创建一个 LXNetClient 对象
Connect	连接指定的服务器端口
GetLocal	得到自身的连接地址(连接后有效)
GetRemote	得到远端的连接地址(连接后有效)
Send	发送一个字符串
Close	关闭当前连接
IsConnect	当前是否已建立连接

new

创建一个 LXNetClient 对象

```
new()
```

参数

无

返回值

类型	描述

LXNetClient	LXNetClient 对象

注意事项

无

示例

```
client = LXNetClient.new()
```

参见

Samples\Test_LuaBase\Test_LXNetClient\脚本\XMain.lua

Connect

连接指定服务器端口

```
Connect(serverip, port)
```

参数

参数名	类型	描述	是否可选	默认值
serverip	string	服务器 IP 地址	否	无
port	number	服务器端口	否	无

返回值

类型	描述
number	错误号： =0 成功 =1 内部创建线程或事件错 =2 超时错 =3 外部关闭了 Socket =4 连接错 =5 参数错 =6 上一次连接未关闭 =7 监听错

注意事项

无

示例

```
local nErr = client.Connect("127.0.0.1",2005)
if 0 ~= nErr then
	XMessageBox("client connect Erro")
end
```

参见

```
Samples\Test_LuaBase\Test_LXNetClient\脚本\XMain.lua
```

GetLocal

得到自身的连接地址(连接后有效)

```
GetLocal()
```

参数

```
无
```

返回值

类型	描述
table	table.ip 表示 ip 地址　table.port 表示端口

注意事项

```
无
```

示例

```
local tLocalInfo = client.GetLocal()
print(tLocalInfo)
```

参见

```
Samples\Test_LuaBase\Test_LXNetClient\脚本\XMain.lua
```

GetRemote

得到远端的连接地址(连接后有效)

```
GetRemote()
```

参数

```
无
```

返回值

类型	描述
table	table.ip 表示 ip 地址　table.port 表示端口

注意事项

无

示例

```
local tRemoteInfo = client.GetRemote()
print(tRemoteInfo)
```

参见

Samples\Test_LuaBase\Test_LXNetClient\脚本\XMain.lua

Send

发送一个字符串

```
Send(strContent, nStrContentLen)
```

参数

参数名	类型	描述	是否可选	默认值
strContent	string	发送内容	否	无
nStrContentLen	number	长度	是	strContent 长度

返回值

成功返回发送的字节数,失败返回 nil

注意事项

strContent 以”\r\n”结尾

示例

```
Local strMsg = "This is a message send by client\r\n"
    Local nSendDataLen = client.Send(strMsg, #strMsg)
if nSendDataLen == nil then
    XMessageBox("Send Data Failed")
else
    print("Send Data, Len = "..tostring(nSendDataLen))
end
```

参见

Samples\Test_LuaBase\Test_LXNetClient\脚本\XMain.lua

Close

关闭当前连接

```
Close
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
client.Close()
```

参见

```
Samples\Test_LuaBase\Test_LXNetClient\脚本\XMain.lua
```

IsConnect

当前是否已建立连接

```
IsConnect()
```

参数

```
无
```

返回值

```
已建立连接返回非 nil  否则返回 nil
```

注意事项

```

```

示例

```
if client.IsConnect() then
end
```

参见

```
Samples\Test_LuaBase\Test_LXNetClient\脚本\XMain.lua
```

3.5 窗口 API

LXWnd

功能描述

该对象实现窗口的各种操作。

函数列表

函数名称	说明
CreateWindow	创建窗口
DestroyWindow	销毁窗口
ActiveWindow	激活当前窗口
AdjustPositionFromParent	根据父窗口调整自身在屏幕中的位置
BringWindowToBottom	把窗口带到最下层
BringWindowToBottomMost	把窗口带到最底层
BringWindowToTop	把窗口带到最上层，如果是 popup 窗口同激活 如果是子窗口将把自身放在 Z order 上面
BringWindowToTopMost	把窗口带到最顶层,仅适用于父窗口是 NULL(虚拟桌面)的 pop 类型窗口
DrawFocusRect	绘制焦点虚框
EnableBorder	有效/无效窗口边框
EnableSize	窗口是否可以改变大小
EnableLinearRender	窗口是否开启线性采样输出
EnableWindow	设置窗口 Enable 或 disable
GetAppScreenMainWnd	得到虚拟窗口句柄
GetBKColor	得到窗口背景色
GetBorderColor	得到窗口边框颜色
GetBorderWidth	得到窗口边框宽度
GetCapture	得到当前捕获鼠标的窗口
GetFont	得到窗口字体
GetFrontColor	得到窗口前景色
GetClientRect	得到窗口客户区矩形
GetCursorPos	得到窗口光标位置
GetDlgItem	得到子控件
GetScreenRect	得到窗口相对于虚拟屏幕的矩形
GetWindowID	得到窗口的 ID
GetWindowRect	得到窗口相对于父窗口矩形
GetWindowData	得到窗口附加数据
GetWindowStyle	得到窗口风格

GetWindowText	得到窗口标题文本
GetHScrollInfo	得到水平滚动条信息
GetImage	得到窗口指定类型图片
GetLightColor	得到窗口高亮色
GetLinearRender	得到是否开启线性采样输出
GetParent	得到父窗口
GetScreenWindowRect	得到虚拟屏幕矩形
GetScrollPos	得到指定类型滚动条当前位置
GetType	得到窗口类型
GetUseParentFont	得到窗口是否使用父窗口字体
GetVScrollInfo	得到窗口垂直滚动条信息
ClientToScreen	窗口客户区坐标转换为虚拟屏幕坐标
WindowToScreen	窗口坐标转换为虚拟屏幕坐标
InvalidateRect	窗口重绘
IsWindowCapture	窗口是否捕获鼠标
IsWindowChild	窗口是否是子窗口
IsClipChildren	窗口是否裁剪孩子
IsClipSibings	窗口是否裁剪兄弟
IsWindowEnabled	窗口是否可用
IsForceUpdate	窗口是否强制刷新
IsWindowActive	窗口是否激活
IsWindowFocus	窗口是否有焦点
IsWindowTransparent	窗口是否透明
IsWindowVisible	窗口是否可见
IsHaveBorder	窗口是否具有边框
IsHaveCaption	窗口是否具有标题栏
IsHaveHScrollBar	窗口是否具有水平滚动条
IsHaveMaximizeBox	窗口是否具有最大化按钮
IsHaveMinimizeBox	窗口是否具有最小化按钮
IsHaveSizeBox	窗口是否可以改变大小
IsHaveSystemMenu	窗口是否具有系统菜单
IsHaveVScrollBar	窗口是否具有垂直滚动条
IsChild	指定窗口是否是本窗口的子窗口
IsOwn	指定窗口是否是本窗口的隶属窗口
IsNoActive	窗口是否无需激活
IsNoFocus	窗口是否无需焦点
IsPaintOnSize	窗口大小变化时是否重绘内容
IsPopMenu	是否为菜单窗口
IsWindowPopup	是否为弹出式窗口
IsWindowValidate	窗口是否有效
KillHotKey	删除一个已创建的窗口热键
KillTimer	删除一个已创建的窗口定时器

LoadScriptCode	从代码串加载 Lua 脚本
LoadScriptFile	从文件加载 Lua 脚本
ModifyStyle	修改窗口风格
MoveWindow	移动窗口
PostMessage	向指定窗口投递一条消息
PtInWindow	判断一个点是否位于窗口内部
ReinvalidateWindows	重新无效窗口及其子窗口
ReleaseCapture	释放捕获的鼠标
ScreenToClient	虚拟屏幕坐标转换为窗口客户区坐标
ScreenToWindow	虚拟屏幕坐标转换为窗口坐标
SendNotifyMessage	向指定窗口发送一条通知消息
SetBKColor	设置窗口背景色
SetBorderColor	设置窗口边框色
SetBorderWidth	设置窗口边框宽度
SetFont	设置窗口字体
SetFrontColor	设置窗口前景色
SetFocus	设置焦点
SetHotKey	设置热键
SetTimer	设置定时器
SetWindowID	设置窗口 ID
SetCapture	捕获鼠标
SetWindowText	设置窗口标题
SetHScrollInfo	设置窗口水平滚动条信息
SetImage	设置窗口指定类型图片
SetLightColor	设置窗口高光颜色
SetLimitInScreen	设置窗口是否受虚拟窗口移动限制
SetScreenLimitWidth	设置窗口移出屏幕的限制
SetScrollPos	设置滚动条位置
SetUseParentFont	设置窗口是否使用父窗口字体
SetVScrollInfo	设置垂直滚动条信息
SetWindowAlpha	设置窗口的透明度
SetCursor	设置窗口的光标
ShowWindow	显示/隐藏窗口
ValidateRect	指定窗口中的一个有效区域
WindowCanClip	窗口是否可以裁减

CreateWindow

创建窗口

```
CreateWindow (pszName, Style, x, y, nWidth, nHeight, pWndParent，nID)
```

参数

参数名	类型	描述	是否可选	默认值
pszName	string	窗口类型	否	无
Style	string	窗口风格	否	无
x	number	窗口左上角 x 值	否	无
y	number	窗口左上角 y 值	否	无
nWidth	number	窗口宽度	否	无
nHeight	number	窗口高度	否	无
pWndParent	窗口对象	父窗口	具体情况再讨论	具体情况再讨论
nID	number	窗口 ID	是	0

返回值

类型	描述
窗口对象	创建成功返回一个有效的窗口对象 ,否则返回 nil

注意事项

Style 参数取值：

系统认为一个字符代表一个风格，一种方法是写入由一系列字符组合而成的字符串，比如’c’代表孩子风格，‘v’代表可见风格，那么 Style 参数就可以是”vc”。其余字符意义在下表中一一列出。请注意，字母不分大小写。

另一种方法是直接使用系统默认宏，系统默认宏含有特定意义，如下将一一列出。需要注意的是，系统默认宏只是一些常用风格，若有特殊要求，必须使用前种方法。

1. 字符代表意义：

控件类型	Style
LXWnd	'C': 孩子窗口，不能和'P'共用
	'O': 是否包含标题栏，弹出窗口有效
	'P': 弹出窗口，不能和'C'共用
	'V': 窗口是否可见
	'L': 是否裁减孩子窗口占有区域
	'B': 是否包含边框
	'S': 是否有系统菜单
	'X': 是否裁剪兄弟窗口
	'T': 窗口是否透明
	'D': 窗口是否禁止
	'F': 窗口是否不接受焦点;
	'G': 窗口是否接收 Tab 键
	'N': 窗口是否无需激活
	'Z': 窗口是否可改变大小，弹出窗口有效
	'M': 窗口是否拥有系统菜单，弹出窗口有效
	'E': 是否拥有垂直滚动杠
	'H': 是否拥有水平滚动杠

	'Q': 改变窗口大小时是否绘制窗口内容，弹出窗口有效
	'A': 是否拥有极大/恢复按钮，弹出窗口有效
	'I': 是否拥有极小化按钮，弹出窗口有效
LXButton	'J': PUSH 按钮
	'R': RAIDO 按钮
	'K': CHECKBOX 按钮
LXComboBox	'J': 简单风格;
	'K': 下拉风格，可编辑
	'Y': 下拉列表风格，不能编辑
LXEdit	'W': 密码风格
	'R': 只读
	'K': 显示键盘按下值，如按下 Ctrl+F，则显示 Ctrl + F
LXGroupBox	无
LXHeadCtrl	无
LXListBox	'J': 多选风格
	'K': 排序
LXListCtrl	'R': 报表风格
	'K': 图标风格
	'Y': 整行选中
LXMenu	'Y': 含竖线
	'W': 不含竖线，但文字与左边界仍然有一定距离
LXProgress	'W': 水平风格
	'R': 垂直风格
LXStatic	'W': 文字
	'K': 图像
	'U': 数字
LXSliderCtrl	与 LXProgress 相同
LXTabCtrl	无
LXTooltip	无
LXTreeCtrl	'K': 含 CheckBox
	'W': 含按钮
	'Y': 含竖线
	'U': 不能拖拽
LXDialog	见 LXWnd

2 系统默认宏的意义：

控件类型	Style
LXWnd	**XWS_POPUP** 包含属性：弹出窗口，可见。
	XWS_CHILD 包含属性：孩子窗口，可见，含边框。
	XWS_POPUPWND, 包含属性：弹出窗口，可见，含标题栏，边框，系统菜单，有“极大/恢复按钮”，有“拥有极小化按钮”。
LXButton	**XWS_PUSHBUTTON**,包含属性： 按钮，孩子窗口，可见，含边框。
	XWS_CHECKBOX_BUTTON,包含属性： CHECKBOX 按钮，孩子窗口，可见，含边框。

	XWS_RADIO,包含属性： Radio 按钮，孩子窗口，可见，含边框。
LXComboBox	**XWS_SIMPLE**,包含属性： 简单风格，孩子窗口，可见，含边框。 **XWS_DROPDOWN**,包含属性： 下拉风格，可编辑，孩子窗口，可见，含边框。 **XWS_ DROPLIST**,包含属性： 下拉列表风格，不能编辑，孩子窗口，可见，含边框。
LXEdit	**XWS_PASSWORD**,包含属性： 密码风格，孩子窗口，可见，含边框。 **XWS_READONLY**,包含属性： 只读风格，孩子窗口，可见，含边框。
LXGroupBox	无
LXHeadCtrl	无
LXListBox	**XWS_MULTISEL**,包含属性： 多选风格，孩子窗口，可见，含边框。 **XWS_LISTSORT**,包含属性： 排序，孩子窗口，可见，含边框。
LXListCtrl	**XWS_REPORT**,包含属性： 报表风格，孩子窗口，可见，含边框。 **XWS_ICON**,包含属性： 图标风格，孩子窗口，可见，含边框。 **XWS_ROWSELECT**,包含属性： 整行选中，孩子窗口，可见，含边框。
LXMenu	**XWS_VERTLINE**,包含属性： 含竖线，弹出窗口，可见，含边框 。 **XWS_LEFTSPACE**,包含属性： 不含竖线，但文字与左边界有一定距离 ，弹出窗口，可见，含边框。
LXProgress	**XWS_VERT**,包含属性： 垂直风格，孩子窗口，可见，含边框。 **XWS_HORZ**,包含属性： 水平风格，孩子窗口，可见，含边框。
LXStatic	**XWS_TEXT**,包含属性： 文字风格，孩子窗口，可见，含边框。 **XWS_IMAGE**,包含属性： 图像风格，孩子窗口，可见，含边框。 **XWS_NUMBER**,包含属性： 数字风格，孩子窗口，可见，含边框。
LXSliderCtrl	**XWS_VERT**,包含属性： 垂直风格，孩子窗口，可见，含边框。 **XWS_HORZ**,包含属性： 水平风格，孩子窗口，可见，含边框。
LXTabCtrl	无
LXTooltip	无
LXTreeCtrl	**XWS_CHECKBOX_TREE**,包含属性： 含 CheckBox，孩子窗口，可见，含边框。 **XWS_BUTTON_TREE**,包含属性： 含按钮，孩子窗口，可见，含边框。 **XWS_LINE**,包含属性： 含竖线，孩子窗口，可见，含边框。 **XWS_NODROP**,包含属性：不能拖拽，孩子窗口，可见，含边框。
LXDialog	与 LXWnd 相同

示例

```
btn = LXButton.new()
--caption, style, x, y, width, height, parent(可选), id(可选)
btn.CreateWindow("TestButton", "cv", 0, 0, 200,20, dlg)
myComboBox = LXComboBox.new()
myComboBox.CreateWindow("Test", XWS_SIMPLE, 0,0,100,20, dlg, 101)
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

DestroyWindow

销毁非模态窗口

```
DestroyWindow ()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
不能销毁模态窗口
```

示例

```
myComboBox. DestroyWindow()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

ActiveWindow

激活/不激活窗口

```
ActiveWindow(bRedrawWindow)
```

参数

参数名	类型	描述	是否可选	默认值
bRedrawWindow	boolean	是否重绘	是	true

返回值

```
无
```

注意事项

无

示例

```
wnd.ActiveWindow(true)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

AdjustPositionFromParent

根据父窗口调整自身在屏幕中的位置

```
AdjustPositionFromParent()
```

参数

无

返回值

无

注意事项

无

示例

```
wnd.AdjustPositionFromParent()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

BringWindowToBottom

使窗口置底

```
BringWindowToBottom(bReDraw)
```

参数

参数名	类型	描述	是否可选	默认值
bReDraw	boolean	是否重绘	是	true

返回值

无

注意事项

无

示例

```
wnd.BringWindowToBottom ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

BringWindowToBottomMost

使窗口置底（仅适用于父窗口为 nil 的 popup 类型窗口）

```
BringWindowToBottomMost(bBottomMost, bReDraw)
```

参数

参数名	类型	描述	是否可选	默认值
bBottomMost	boolean	是否置底	是	true
bReDraw	boolean	是否重绘	是	true

返回值

无

注意事项

无

示例

```
wnd.BringWindowToBottomMost()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

BringWindowToTop

使窗口置顶

```
BringWindowToTop(bReDraw)
```

参数

参数名	类型	描述	是否可选	默认值

bReDraw	boolean	是否重绘	是	true

返回值

无

注意事项

无

示例

```
wnd.BringWindowToTop()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

BringWindowToTopMost

使窗口置顶（仅适用于父窗口为 nil 的 popup 类型窗口）

```
BringWindowToTop(bTopMost, bReDraw)
```

参数

参数名	类型	描述	是否可选	默认值
bTopMost	boolean	是否置顶	是	true
bReDraw	boolean	是否重绘	是	true

返回值

无

注意事项

无

示例

```
wnd.BringWindowToTopMost()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

DrawFocusRect

画焦点虚框矩形区域

```
DrawFocusRect(rect)或
DrawFocusRect(left, top ,right, bottom)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	目标矩形区域	否	无
left	number	左上角 X 坐标	否	无
top	number	左上角 Y 坐标	否	无
right	number	右下角 X 坐标	否	无
bottom	number	右下角 Y 坐标	否	无

返回值

无

注意事项

矩形区域为空时，则取消先前的绘制

示例

```
Rect = LXRect.new(0,0,100,100)
wnd.DrawFocusRect(Rect)
或
wnd.DrawFocusRect(0,0,100,100)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

EnableBorder

使能/禁止边框

```
EnableBorder(bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否有边框	是	true

返回值

无

注意事项

无

示例

```
wnd.EnableBorder()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

EnableSize

使能/禁止改变大小

```
EnableSize(bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否能改变大小	是	true

返回值

```
无
```

注意事项

```
无
```

示例

```
wnd. EnableSize ()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

EnableLinearRender

使能/禁止线性采样输出

```
EnableLinearRender(bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否线性采样输出	否	无

返回值

```
无
```

注意事项

无

示例

```
wnd.EnableLinearRender (true)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

EnableWindow

设置使窗口 Enable/Disable

```
EnableWindow(bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否窗口能用	是	true

返回值

无

注意事项

无

示例

```
wnd.EnableWindow()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetAppScreenMainWnd

得到虚拟屏幕的窗口句柄（Windows 真实句柄）

```
GetAppScreenMainWnd()
```

参数

无

返回值

类型	描述
number	窗口句柄

注意事项

无

示例

```
hwnd = wnd.GetAppScreenMainWnd()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetBKColor

得到窗口背景色

```
GetBKColor()
```

参数

无

返回值

类型	描述
LXColor	背景颜色

注意事项

无

示例

```
color = wnd.GetBKColor()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetBorderColor

得到窗口边框色

```
GetBorderColor()
```

参数

无

返回值

类型	描述
LXColor	边框颜色

注意事项

无

示例

```
color = wnd.GetBorderColor()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetBorderWidth

得到窗口边框宽度

```
GetBorderWidth()
```

参数

无

返回值

类型	描述
number	边框宽度

注意事项

无

示例

```
Width = wnd.GetBorderWidth()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetCapture

得到当前捕获鼠标的窗口

GetCapture ()

参数

无

返回值

类型	描述
LXWnd	捕获鼠标的窗口

注意事项

无

示例

```
CaptureWnd = wnd.GetCapture ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetFont

得到窗口字体

GetFont()

参数

无

返回值

类型	描述
LXFont	窗口字体

注意事项

无

示例

```
font = wnd.GetFont()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetFrontColor

得到窗口前景色

```
GetFrontColor()
```

参数

无

返回值

类型	描述
LXColor	窗口前景色

注意事项

无

示例

```
color = wnd.GetFrontColor()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetClientRect

得到窗口客户区

```
GetClientRect()
```

参数

无

返回值

类型	描述
LXRect	窗口客户区矩形区域

注意事项

无

示例

```
rect = wnd.GetClientRect()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetCursorPos

得到窗口光标位置

```
GetCursorPos()
```

参数

无

返回值

类型	描述
LXPoint	窗口光标位置

注意事项

无

示例

```
point = wnd.GetCursorPos()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetDlgItem

通过对话框控件 ID 得到对话框子控件对象

```
GetDlgItem(ID)
```

参数

参数名	类型	描述	是否可选	默认值
ID	number	控件 ID	否	无

返回值

类型	描述
控件对象，比如 LXButton 等等类型	控件对象

注意事项

无

示例

无

参见

Samples\Test_Wnd\Test_LXDialog\脚本\XMain.lua

GetScreenRect

得到窗口矩形，相对于虚拟屏幕

```
GetScreenRect ()
```

参数

无

返回值

类型	描述
LXRect	窗口虚拟屏幕区域

注意事项

无

示例

```
rect = wnd. GetScreenRect ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetWindowID

得到窗口 ID

```
GetWindowID()
```

参数

无

返回值

类型	描述
number	窗口 ID

注意事项

无

示例

```
ID = wnd. GetWindowID ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetWindowRect

得到窗口相对于父窗口矩形区域

```
GetWindowRect()
```

参数

无

返回值

类型	描述
LXRect	相对于父窗口的矩形区域

注意事项

无

示例

```
rect = wnd. GetWindowRect ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetWindowData

得到窗口附加数据

```
GetWindowData ()
```

参数

无

返回值

类型	描述
number	窗口附加数据

注意事项

无

示例

```
Data = wnd. GetWindowData ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetWindowStyle

得到窗口风格

```
GetWindowStyle ()
```

参数

无

返回值

类型	描述
string	窗口风格

注意事项

无

示例

```
Style = wnd. GetWindowStyle ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetWindowText

得到窗口标题文本

```
GetWindowText ()
```

参数

无

返回值

类型	描述
string	窗口文本

注意事项

无

示例

```
Text = wnd. GetWindowText ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetHScrollInfo

得到水平滚动条信息

```
GetHScrollInfo()
```

参数

无

返回值

类型	描述
number	水平滚动条总大小
number	每一页大小
number	每一列大小

注意事项

无

示例

```
TotalSize, PageSize, LineSize = wnd.GetHScrollInfo()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetImage

根据指定类型得到窗口图片和绘制方式

```
GetImage(type)
```

参数

参数名	类型	描述	是否可选	默认值
type	number	图片类型	否	无

返回值

类型	描述
LXHandle	图片句柄
number	绘制方式

注意事项

type, number 的值的意义请参考 SetImage 函数

示例

```
ImgHandle, DrawStyle = wnd.GetImage()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetLightColor

得到窗口高亮色

```
GetLightColor()
```

参数

无

返回值

类型	描述
LXColor	高亮颜色

注意事项

无

示例

```
color = wnd.GetLightColor()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetLinearRender

得到线性采样输出状态

```
GetLinearRender()
```

参数

无

返回值

类型	描述
boolean	是否应用线性采样输出状态

注意事项

无

示例

```
bEnable = wnd.GetLinearRender()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

GetParent

获取父窗口

```
GetParent()
```

参数

```
无
```

返回值

类型	描述
LXWnd	父窗口

注意事项

```
无
```

示例

```
hwndParent = wnd.GetParent()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

GetScreenWindowRect

得到虚拟屏幕矩形

```
GetScreenWindowRect ()
```

参数

```
无
```

返回值

类型	描述
LXRect	虚拟屏幕矩形区域

注意事项

```
无
```

示例

```
rect = wnd. GetScreenWindowRect ()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

GetScrollPos

得到指定类型滚动条位置

```
GetScrollPos(type)
```

参数

参数名	类型	描述	是否可选	默认值
type	number	滚动条类型	是	0

返回值

类型	描述
number	滚动条位置

注意事项

ScrollBar 类型 type 取值如下：
= 0　　水平滚动条
= 1　　垂直滚动条

示例

```
Pos = wnd.GetScrollPos()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetType

得到窗口类型

```
GetType()
```

参数

无

返回值

类型	描述
string	窗口类型

注意事项

无

示例

```
Type = wnd.GetType()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetUseParentFont

得到窗口是否使用父窗口字体

GetUseParentFont ()

参数

无

返回值

类型	描述
boolean	窗口是否使用父窗口字体

注意事项

无

示例

bUse = wnd.GetUseParentFont ()

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

GetVScrollInfo

得到垂直滚动条信息

GetVScrollInfo()

参数

无

返回值

类型	描述
number	水平滚动条总大小
number	每一页大小
number	每一行大小

注意事项

无

示例

TotalSize, PageSize, LineSize = wnd.GetVScrollInfo()

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ClientToScreen

窗口客户区坐标转换成虚拟屏幕坐标

ClientToScreen(point)
ClientToScreen(rect)

参数

参数名	类型	描述	是否可选	默认值

point	LXPoint	要转换的坐标点	否	无
rect	LXRect	要转换的矩形区域	否	无

返回值

无

注意事项

转化后的值已经在传进来的参数里面

示例

```
Point = LXPoint.new(100, 100)
wnd. ClientToScreen(Point)
Rect = LXRect.new(0, 0, 100, 100)
wnd. ClientToScreen(Rect)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

WindowToScreen

窗口坐标转换成虚拟屏幕坐标

```
WindowToScreen (point)
WindowToScreen (rect)
```

参数

参数名	类型	描述	是否可选	默认值
point	LXPoint	要转换的坐标点	否	无
rect	LXRect	要转换的矩形区域	否	无

返回值

无

注意事项

转化后的值已经在传进来的参数里面

示例

```
Point = LXPoint.new(100, 100)
wnd. WindowToScreen (Point)
Rect = LXRect.new(0, 0, 100, 100)
wnd. WindowToScreen (Rect)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

InvalidateRect

使窗口无效进行重绘

```
InvalidateRect (rect,bIncludeNC)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	指定矩形范围，忽略则重绘整个窗口	是	nil
bIncludeNC	boolean	如果 bIncludeNC = true rect 相对于窗口坐标否则 rect 相对于窗口客户区坐标	是	true

返回值

无

注意事项

无

示例

```
wnd. InvalidateRect ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsCapture

窗口是否捕获鼠标

```
IsCapture()
```

参数

无

返回值

类型	描述
boolean	是否捕获鼠标

注意事项

无

示例

```
bCapture = wnd.IsCapture()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsWindowChild

窗口是否为子窗口

```
IsWindowChild ()
```

参数

无

返回值

类型	描述
boolean	是否为子窗口

注意事项

无

示例

```
bChild = wnd. IsWindowChild ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsClipChildren

窗口是否裁剪子窗口

```
IsClipChildren()
```

参数

无

返回值

类型	描述
boolean	是否裁剪子窗口

注意事项

无

示例

```
bClip = wnd.IsClipChildren()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsClipSibings

窗口是否裁剪兄弟窗口

```
IsClipSibings()
```

参数

无

返回值

类型	描述
boolean	是否裁剪兄弟窗口

注意事项

无

示例

```
bClip = wnd.IsClipSibings()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsWindowEnabled

获取窗口是否可用

```
IsWindowEnabled ()
```

参数

无

返回值

类型	描述
boolean	窗口是否可用

注意事项

无

示例

```
bEnable = wnd. IsWindowEnabled ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsForceUpdate

获取窗口是否需要强制更新

```
IsForceUpdate()
```

参数

无

返回值

类型	描述
boolean	窗口是否需要强制更新

注意事项

无

示例

```
bUpdate = wnd.IsForceUpdate()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsWindowActive

获取窗口是否激活

```
IsWindowActive ()
```

参数

```
无
```

返回值

类型	描述
boolean	窗口是否激活

注意事项

```
无
```

示例

```
bActive = wnd. IsWindowActive ()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

IsWindowFocus

获取窗口是否具有焦点

```
IsWindowFocus ()
```

参数

```
无
```

返回值

类型	描述
boolean	窗口是否具有焦点

注意事项

```
无
```

示例

```
bFocus = wnd. IsWindowFocus ()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

IsWindowTransparent

获取窗口是否透明

```
IsWindowTransparent ()
```

参数

```
无
```

返回值

类型	描述
boolean	窗口是否透明

注意事项

无

示例

```
bTrans = wnd. IsWindowTransparent ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsWindowVisible

获取窗口是否可见

```
IsWindowVisible ()
```

参数

无

返回值

bVisible	boolean	窗口是否可见

注意事项

无

示例

```
bVisible = wnd. IsWindowVisible ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsHaveBorder

窗口是否具有边框

```
IsHaveBorder()
```

参数

无

返回值

bHave	boolean	是否具有边框

注意事项

无

示例

```
bHave = wnd.IsHaveBorder()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsHaveCaption

窗口是否具有标题栏

```
IsHaveCaption()
```

参数

无

返回值

bHave	boolean	是否具有标题栏

注意事项

无

示例

```
bHave = wnd.IsHaveCaption()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsHaveHScrollBar

窗口是否具有水平滚动条

```
IsHaveHScrollBar()
```

参数

无

返回值

类型	描述
boolean	是否有水平滚动条

注意事项

无

示例

```
bHave = wnd.IsHaveHScrollBar()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsHaveMaximizeBox

窗口是否具有最大化按钮

```
IsHaveMaximizeBox()
```

参数

无

返回值

类型	描述
boolean	是否有最大化按钮

注意事项

无

示例

```
bHave = wnd.IsHaveMaximizeBox()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsHaveMinimizeBox

窗口是否具有最小化按钮

```
IsHaveMinimizeBox()
```

参数

无

返回值

类型	描述
boolean	是否有最小化按钮

注意事项

无

示例

```
bHave = wnd.IsHaveMinimizeBox()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsHaveSizeBox

窗口是否可以改变大小

```
IsHaveSizeBox()
```

参数

无

返回值

类型	描述
boolean	是否可以改变大小

注意事项

无

示例

```
bHave = wnd.IsHaveSizeBox()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

IsHaveSystemMenu

窗口是否具有系统菜单

```
IsHaveSystemMenu()
```

参数

无

返回值

类型	描述
boolean	是否有系统菜单

注意事项

无

示例

```
bHave = wnd.IsHaveSystemMenu()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

IsHaveVScrollBar

窗口是否具有垂直滚动条

```
IsHaveVScrollBar()
```

参数

无

返回值

类型	描述
boolean	是否有垂直滚动条

注意事项

无

示例

```
bHave = wnd.IsHaveVScrollBar()
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

IsChild

窗口是否是自己的子窗口

IsChild(ChildWnd)

参数

参数名	类型	描述	是否可选	默认值
ChildWnd	LXWnd	窗口	否	无

返回值

类型	描述
boolean	是否是自己的子窗口

注意事项

无

示例

```
bChild = wnd.IsChild(ChildWnd)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsOwn

窗口是否是自己的隶属窗口

IsOwn (ChildWnd)

参数

参数名	类型	描述	是否可选	默认值
ChildWnd	LXWnd	窗口	否	无

返回值

类型	描述
boolean	是否是自己的隶属窗口

注意事项

无

示例

```
bOwn = wnd.IsOwn (ChildWnd)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsNoActive

窗口是否无需激活

IsNoActive()

参数

无

返回值

类型	描述
boolean	是否无需激活

注意事项

无

示例

```
bNoActive = wnd.IsNoActive()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsNoFocus

窗口是否无需焦点

```
IsNoFocus()
```

参数

无

返回值

类型	描述
boolean	是否无需焦点

注意事项

无

示例

```
bNoFocus = wnd.IsNoFocus()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsPaintOnSize

改变窗口大小时是否显示窗口内容

```
IsPaintOnSize()
```

参数

无

返回值

类型	描述
boolean	改变大小是是否显示窗口内容

注意事项

无

示例

```
bPaint = wnd.IsPaintOnSize()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsPopMenu

是否为菜单窗口

```
IsPopMenu()
```

参数

无

返回值

类型	描述
boolean	是否为菜单窗口

注意事项

无

示例

```
bPopMenu = wnd.IsPopMenu()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsWindowPopup

是否为弹出式窗口

```
IsWindowPopup ()
```

参数

无

返回值

类型	描述
boolean	是否为弹出窗口

注意事项

无

示例

```
bPopWnd = wnd. IsWindowPopup ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

IsWindowValidate

窗口是否有效

```
IsWindowValidate ()
```

参数

无

返回值

类型	描述
boolean	是否窗口有效

注意事项

无

示例

```
bValidate = wnd. IsWindowValidate ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

KillHotKey

删除指定的热键

```
KillHotKey(Key)
```

参数

参数名	类型	描述	是否可选	默认值
Key	number	热键 ID，该值为 SetHotkey 返回值	否	无

返回值

类型	描述
boolean	是否删除成功

注意事项

无

示例

```
bRes = wnd.KillHotKey(1000)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

KillTimer

删除指定的定时器

```
KillTimer(Timer)
```

参数

参数名	类型	描述	是否可选	默认值
Timer	number	定时器 ID，该值必须和 SetTimer 的第一个参数相同	否	无

返回值

类型	描述
boolean	是否删除成功

注意事项

无

示例

```
bRes = wnd.KillTimer(1000)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

LoadScriptCode

从代码串加载 Lua 脚本

```
LoadScriptCode(Code)
```

参数

参数名	类型	描述	是否可选	默认值
Code	string	代码段	否	无

返回值

类型	描述
boolean	装载是否成功

注意事项

无

示例

```
bRes = wnd.LoadScriptFile(“print(“haha”)”)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

LoadScriptFile

从文件加载 Lua 脚本

```
LoadScriptFile(FileName)
```

参数

参数名	类型	描述	是否可选	默认值
FileName	string	脚本文件名	否	无

返回值

类型	描述
boolean	装载是否成功

注意事项

无

示例

```
bRes = wnd.LoadScriptFile(“D:\\1.lua”)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ModifyStyle

更改窗口风格

```
ModifyStyle(AddStyle, DelStyle)
```

参数

参数名	类型	描述	是否可选	默认值
AddStyle	string	增加的风格	否	无
DelStyle	string	删除的风格	是	nil

返回值

无

注意事项

AddStyle 和 DelStyle 的取值与 CreateWindow 函数的 Style 相同，请参考

示例

```
wnd.ModifyStyle(“a”)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

MoveWindow

移动窗口

```
MoveWindow (l, t, r, b, bRedraw)
MoveWindow (rc, bRedraw)
```

参数

参数名	类型	描述	是否可选	默认值

l	number	左上角 X 坐标	否	无
t	number	左上角 Y 坐标	否	无
r	number	右下角 X 坐标	否	无
b	number	右下角 Y 坐标	否	无
bRedraw	boolean	bRedraw	有	true
rc	LXRect	矩形区域	否	无

返回值

无

注意事项

只有窗口改变了大小并且 bRedrawWindow 为 TRUE,才刷新窗口

示例

```
rc = LXRect.new(100, 100, 300,300, 1)
wnd. MoveWindow (rc)
wnd. MoveWindow (100, 100, 200, 200, true)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

PostMessage

移动窗口

```
PostMessage (pWnd, nNotifyID, nWparam, nLparam)
```

参数

参数名	类型	描述	是否可选	默认值
pWnd	窗口对象	要接收消息的窗口	否	无
nNotifyID	number	消息 ID	否	无
nWparam	number	参数一	是	nil
nLparam	number	参数一	是	nil

返回值

类型	描述
boolean	如果处理了该通知请返回 TRUE　未处理返回 FALSE

注意事项

示例

```
wnd. PostMessage (wnd, X_ONBUTTONCLICK, nil,nil)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

PtInWindow

判断一个点是否在窗口里面

```
PtInWindow (point)
PtInWindow (x,y)
```

参数

参数名	类型	描述	是否可选	默认值
point	LXPoint	测试点	否	无
x	number	测试点的 Y 坐标	否	无
y	number	测试点的 X 坐标	否	无

返回值

类型	描述
boolean	点是否在窗口内

注意事项

无

示例

```
bInWindow = wnd. PtInWindow (20, 30)
pt = LXPoint.new(30, 50)
bInWindow = wnd. PtInWindow (pt)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ReinvalidateWindows

重新无效窗口及其子窗口

```
ReinvalidateWindows ()
```

参数

无

返回值

类型	描述
boolean	重绘是否成功

注意事项

无

示例

```
bRes = wnd. ReinvalidateWindows ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ReleaseCapture

释放捕获的鼠标

```
ReleaseCapture ()
```

参数

无

返回值

类型	描述
boolean	释放是否成功

注意事项

无

示例

```
bRes = wnd. ReleaseCapture ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ScreenToClient

虚拟屏幕坐标转换成窗口客户区坐标

```
ScreenToClient (point)
ScreenToClient (rect)
```

参数

参数名	类型	描述	是否可选	默认值
point	LXPoint	要转换的坐标点	否	无
rect	LXRect	要转换的矩形区域	否	无

返回值

无

注意事项

转化后的值已经在传进来的参数里面

示例

```
Point = LXPoint.new(100, 100)
wnd. ScreenToClient (Point)
Rect = LXRect.new(0, 0, 100, 100)
wnd. ScreenToClient (Rect)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ScreenToWindow

虚拟屏幕坐标转换成窗口客户区坐标

```
ScreenToWindow (point)
ScreenToWindow (rect)
```

参数

参数名	类型	描述	是否可选	默认值
point	LXPoint	要转换的坐标点	否	无
rect	LXRect	要转换的矩形区域	否	无

返回值

无

注意事项

转化后的值已经在传进来的参数里面

示例

```
Point = LXPoint.new(100, 100)
wnd. ScreenToWindow (Point)
Rect = LXRect.new(0, 0, 100, 100)
wnd. ScreenToWindow (Rect)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SendNotifyMessage

向其它窗口发送消息

```
bState = SendNotifyMessage (pWnd, nNotifyID, nNotifyData)
```

参数

参数名	类型	描述	是否可选	默认值
pWnd	窗口对象	要接收消息的窗口	否	无
nNotifyID	number	消息 ID	否	无
nNotifyData	number	消息数据	是	nil

返回值

类型	描述
boolean	如果处理了该通知请返回 TRUE　未处理返回 FALSE

注意事项

示例

```
wnd. SendNotifyMessage(wnd, X_ONBUTTONCLICK, nil,nil)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetBKColor

设置窗口背景色

```
SetBKColor(color)
SetBKColor(r, g, b)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无
r	number	R 值	否	无
g	number	G 值	否	无
b	number	B 值	否	无

返回值

无

注意事项

无

示例

```
color = LXColor.new(255, 0, 0)
wnd.SetBKColor(color)
或
wnd.SetBKColor(255, 0, 0)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetBorderColor

设置窗口边框色

```
SetBorderColor(color)
SetBorderColor (r, g, b)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无
r	number	R 值	否	无
g	number	G 值	否	无
b	number	B 值	否	无

返回值

无

注意事项

无

示例

```
color = LXColor.new(255, 0, 0)
wnd.SetBorderColor(color)
wnd.SetBorderColor(255,0,0)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetBorderWidth

设置窗口边框宽度

```
SetBorderWidth(Width)
```

参数

参数名	类型	描述	是否可选	默认值
Width	number	边框宽度	否	无

返回值

无

注意事项

无

示例

```
wnd.SetBorderWidth(2)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetFont

设置窗口字体

```
SetFont(font, bShare)
```

参数

参数名	类型	描述	是否可选	默认值
font	LXFont	窗口字体	否	无
bShare	boolean	该字体是否共享	是	false

返回值

无

注意事项

无

示例

```
font = LXFont.new()
```

```
font.Create(15, “宋体”)
wnd.SetFont(font)
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

SetFrontColor

设置窗口前景色

```
SetFrontColor(color)
SetBorderColor (r, g, b)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无
r	number	R 值	否	无
g	number	G 值	否	无
b	number	B 值	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
color = LXColor.new(255, 0, 0)
wnd.SetFrontColor(color)
wnd.SetFrontColor(255, 0, 0)
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

SetFocus

设置窗口焦点

```
SetFocus (bFocus)
```

参数

参数名	类型	描述	是否可选	默认值
bFocus	boolean	是否设置焦点	是	true

返回值

```
无
```

注意事项

```
无
```

示例

```
wnd. SetFocus (false)
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

SetHotKey

设置窗口热键

```
SetHotKey (Key, CtrlKey)
```

参数

参数名	类型	描述	是否可选	默认值
Key	number	虚拟健码值	否	无
CtrlKey	number	附加组合键	是	0

返回值

类型	描述
number	创建成功，返回热键 ID，失败返回 0

注意事项

附加键状态 CtrlKey 取值如下：
= 0 不需附加键
= 1 Alt 必须按下
= 2 Ctrl 必须按下
= 3 Shift 必须按下

示例

```
HotKeyID = wnd. SetHotKey (10)
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

SetTimer

设置窗口定时器

```
SetTimer (IDEvent, Elapse)
```

参数

参数名	类型	描述	是否可选	默认值
IDEvent	number	定时器消息 ID	否	无
Elapse	number	定时器消息间隔，单位为毫秒	否	无

返回值

类型	描述
number	创建成功，返回定时器 ID，失败返回 0

注意事项

无

示例

```
HotKeyID = wnd. SetTimer (1000, 400)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetWindowID

设置窗口 ID

```
SetWindowID (ID)
```

参数

参数名	类型	描述	是否可选	默认值
ID	number	窗口 ID	否	无

返回值

无

注意事项

无

示例

```
wnd. SetWindowID (1000)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetCapture

设置窗口捕获鼠标

```
SetCapture ()
```

参数

无

返回值

类型	描述
boolean	设置是否成功

注意事项

无

示例

```
wnd. SetCapture ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetWindowText

设置窗口标题文本

SetWindowText (text)

参数

参数名	类型	描述	是否可选	默认值
text	string	窗口文本	否	无

返回值

无

注意事项

无

示例

wnd. SetWindowText (“Hello”)

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetHScrollInfo

设置窗口水平滚动条信息

SetHScrollInfo(TotalSize, PageSize, LineSize)

参数

参数名	类型	描述	是否可选	默认值
TotalSize	number	总大小	否	无
PageSize	number	每一页大小	是	nil
LineSize	number	每一列大小	是	nil

返回值

无

注意事项

无

示例

wnd.SetHScrollInfo(100, 20, 1)

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetImage

设置窗口指定类型图片

```
SetImage(strImg, Type, DrawStyle)
SetImage(ImgHandle, Type, DrawStyle)
```

参数

参数名	类型	描述	是否可选	默认值
strImg	string	表示图像路径	否	无
ImgHandle	LXHandle	表示图像的 LXHandle 对象	否	无
Type	number	表示绘制图像类型	是	0
DrawStyle	number	表示图像的绘制方式	是	0

返回值

类型	描述
boolean	设置是否成功

注意事项

1. 图片类型 Type 取值如下：
X_IMG_UNKNOWN：未定义的窗口图像类型
X_IMG_NORMAL：正常背景图
X_IMG_BUTTON：按钮图像(Button)
X_IMG_PROGRESSSETP：进度条图像(Progress)
X_IMG_SLIDERSTEP：滑块拖动按钮图像(Slider)
X_IMG_COMBOBOXBTN：ComboBox 按钮图像(ComboBox)
X_IMG_COMBOBOXLIST：ComboBox 弹出 ListBox 背景图像
X_IMG_SCROLLPREVBTN：ScrollBar 上翻按钮
X_IMG_SCROLLNEXTBTN：ScrollBar 下翻按钮
X_IMG_SCROLLDRAGTBTN：ScrollBar 拖动块按钮
2. 绘制方式 DrawStyle 取值如下：
= 0　　拉伸
= 1　　平铺
= 2　　九分图

示例

```
ImgHandle = LXHandle.new(“D:\\BK.jpg”)
wnd.SetImage(ImgHandle)
或者
ComboBox.SetImage(“D:\\BK.jpg”, X_IMG_COMBOBOXBTN, 1)
```

参见

```
Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua
```

SetLightColor

设置窗口高亮色

```
SetLightColor(color) 或
SetLightColor (r, g, b)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无
r	number	R 值	否	无
g	number	G 值	否	无
b	number	B 值	否	无

返回值

无

注意事项

无

示例

```
color = LXColor.new(255, 0, 0)
wnd.SetLightColor(color)
或
wnd.SetLightColor(255, 0, 0)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetLimitInScreen

设置窗口是否只在虚拟屏幕内移动

```
SetLimitInScreen(size)
```

参数

参数名	类型	描述	是否可选	默认值
size	number	设置 Popup 窗口是否受虚拟屏幕的移动限制即不能移动到虚拟屏幕之外以免看不见	是	20

返回值

无

注意事项

无

示例

```
wnd.SetLimitInScreen(20)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetScreenLimitWidth

设置窗口移出虚拟屏幕的限制

```
SetScreenLimitWidth(Width)
```

参数

参数名	类型	描述	是否可选	默认值
Width	number	设置窗口移出虚拟屏幕的限制	是	20

返回值

无

注意事项

无

示例

```
wnd.SetScreenLimitWidth (20)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetScrollPos

设置指定类型 ScrollBar 位置

```
SetScrollPos(Type, Pos)
```

参数

参数名	类型	描述	是否可选	默认值
Type	number	滚动条类型	否	无
Pos	number	滚动条当前位置	否	无

返回值

无

注意事项

ScrollBar 类型 Type 取值如下：
= 0　　水平滚动条
= 1　　垂直滚动条

示例

```
wnd.SetScrollPos(0, 100)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetUseParentFont

设置窗口是否使用父窗口字体

SetUseParentFont (bUse)

参数

参数名	类型	描述	是否可选	默认值
bUse	boolean	窗口是否使用父窗口字体	否	无

返回值

无

注意事项

无

示例

wnd.GetUseParentFont (true)

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetVScrollInfo

设置窗口垂直滚动条信息

SetVScrollInfo(TotalSize, PageSize, LineSize)

参数

参数名	类型	描述	是否可选	默认值
TotalSize	number	总大小	否	无
PageSize	number	每一页大小	是	nil
LineSize	number	每一行大小	是	nil

返回值

无

注意事项

无

示例

wnd.SetVScrollInfo(100, 20, 1)

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetWindowAlpha

设置窗口透明度

SetWindowAlpha(Alpha)

参数

参数名	类型	描述	是否可选	默认值

Alpha	number	窗口透明度，取值范围 0-255, 0 表示完全透明，255 表示完全显示	是	128

返回值

类型	描述
boolean	设置是否成功

注意事项

无

示例

```
bRes = wnd.SetWindowAlpha(100)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

SetCursor

设置窗口光标

```
SetCursor (Cursor)
```

参数

参数名	类型	描述	是否可选	默认值
Cursor	LXCursor	光标句柄	否	无

返回值

类型	描述
boolean	设置是否成功

注意事项

无

示例

```
cursor = LXCursor.new("busy")
bRes = wnd. SetCursor (cursor)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ShowWindow

设置显示/隐藏窗口

```
ShowWindow (bShow, bModal, bRedraw, bActive)
```

参数

参数名	类型	描述	是否可选	默认值
bShow	boolean	是否显示窗口	是	true
bModal	boolean	是否显示模态窗口	是	false
bRedraw	boolean	是否重绘窗口	是	true
bActive	boolean	是否激活窗口	是	true

返回值

类型	描述
boolean	设置是否成功

注意事项

当 LXDialog 对象调用该函数时第二个参数无效

示例

```
bRes = wnd. ShowWindow ()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

ValidateRect

指定一个窗口区域有效，指定有效后，该区域将不进行绘制

```
ValidateRect (rect, bIncludeNC)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	指定矩形范围，忽略则重绘整个窗口	是	nil
bIncludeNC	boolean	如果 bIncludeNC = true, rect 相对于窗口坐标. 否则 rect 相对于窗口客户区坐标	是	true

返回值

无

注意事项

无

示例

```
rect = LXRect.new(0, 0, 100, 100)
wnd. ValidateRect (rect)
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

WindowCanClip

子窗口是否可裁剪

```
WindowCanClip()
```

参数

无

返回值

类型	描述
boolean	子窗口是否可裁剪

注意事项

无

示例

```
bClip = wnd.WindowCanClip()
```

参见

Samples\Test_Wnd\Test_LXWnd\脚本\Main.lua

LXButton

功能描述

该对象实现按钮控件功能。

函数列表

函数名称	说明
SetCheck	设置按钮是否勾中
GetCheck	获取按钮是否勾中

GetCheck

获取密码显示字符

```
GetCheck()
```

参数

无

返回值

类型	描述
boolean	按钮是否勾中

注意事项

无

示例

```
bCheck = button.GetCheck()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXButton\脚本\XMain.lua

SetCheck

设置密码显示字符

```
SetCheck (bCheck)
```

参数

参数名	类型	描述	是否可选	默认值
bCheck	boolean	是否勾中	否	无

返回值

无

注意事项

无

示例

```
button.SetCheck(true)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXButton\脚本\XMain.lua

LXStatic

功能描述

该对象实现静态控件功能。

函数列表

函数名称	说明
SetCharSet	设置字符集
GetCharSet	获得字符集
SetImageInfo	设置数字图片行列信息

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

SetCharSet

设置字符集

```
SetCharSet(str)
```

参数

参数名	类型	描述	是否可选	默认值
str	string	字符集	否	无

返回值

无

注意事项

无

示例

```
--设置数字字符集
static.SetCharSet(“0123456789”)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXStatic\脚本\XMain.lua

GetCharSet

获得字符集

```
GetCharSet(str)
```

参数

无

返回值

类型	描述
string	字符集

注意事项

无

示例

```
--获得字符集
local str = static.GetCharSet()
print(str)          --输出 0123456789
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXStatic\脚本\XMain.lua

SetImageInfo

设置字符集图片的行列信息

```
SetImageInfo(Row, Col)
```

参数

参数名	类型	描述	是否可选	默认值
Row	number	图片的行	否	无
Col	number	图片的列	否	无

返回值

无

注意事项

无

示例

```
--设置数字字符集图片行列信息
static.SetImageInfo(1, 10)                -- “0123456789”是一副 1 行 10 列的图片
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXStatic\脚本\XMain.lua

LXEdit

功能描述

该对象实现文本框控件功能。

函数列表

函数名称	说明
GetPasswordChar	获取密码显示字符
SetPasswordChar	设置密码显示字符
GetLimitText	获取允许输入字符串最大长度
SetLimitText	设置允许输入字符串最大长度
GetSelBkColor	获取选中背景色
SetSelBkColor	设置选中背景色
GetTextColor	获取文本色
SetTextColor	设置文本色
GetSelTextColor	获取选中文本色
SetSelTextColor	设置选中文本色
GetSel	获取选中字符
SetSel	设置选中字符
Copy	拷贝文本框内容
Paste	粘贴文本框内容
Cut	剪切文本框内容
Clear	清除文本框内容
InsertText	在指定位置插入文本

GetPasswordChar

获取密码显示字符

```
GetPasswordChar()
```

参数

无

返回值

类型	描述
string	密码显示字符

注意事项

无

示例

```
strPass = edit.GetPasswordChar()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

SetPasswordChar

设置密码显示字符

```
SetPasswordChar(str)
```

参数

参数名	类型	描述	是否可选	默认值
str	string	密码显示字符	否	无

返回值

无

注意事项

无

示例

```
edit. SetPasswordChar(“a”)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

GetLimitText

获取允许输入字符串最大长度

```
GetLimitText()
```

参数

无

返回值

类型	描述
number	最大字符串长度

注意事项

无

示例

```
nLen = edit.GetLimitText()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

SetLimitText

设置允许输入字符串最大长度

```
SetLimitText(nLen)
```

参数

参数名	类型	描述	是否可选	默认值
nLen	number	最大字符串长度	否	无

返回值

无

注意事项

无

示例

```
edit.SetLimitText(5)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

GetSelBkColor

获取选中背景色

```
GetSelBkColor()
```

参数

无

返回值

类型	描述
LXColor	选中背景色

注意事项

无

示例

```
color = edit.GetSelBkColor()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

SetSelBkColor

设置选中背景色

```
SetSelBkColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	选中背景色	否	无

返回值

无

注意事项

无

示例

```
edit.SetSelBkColor(color)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

GetTextColor

获取文本色

```
GetTextColor()
```

参数

无

返回值

类型	描述
LXColor	文本色

注意事项

无

示例

```
color = edit.GetTextColor()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

SetTextColor

设置文本色

```
SetTextColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	设置文本色	否	无

返回值

无

注意事项

无

示例

```
edit.SetTextColor(color)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

GetSelTextColor

获取选中文本色

```
GetSelTextColor()
```

参数

无

返回值

类型	描述
LXColor	获取选中文本色

注意事项

无

示例

```
color = edit.GetSelTextColor()
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua
```

SetSelTextColor

设置选中文本色

```
SetSelTextColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	设置选中文本色	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
edit.SetSelTextColor(color)
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua
```

GetSel

获取选中字符串

```
GetSel()
```

参数

```
无
```

返回值

类型	描述
string	选中字符串

number	起始字符索引
number	终止字符索引

注意事项

无

示例

```
Str, nStartChar, nEndChar = edit.GetSel()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

SetSel

设置选中字符串

```
SetSel(nStartChar, nEndChar)
```

参数

参数名	类型	描述	是否可选	默认值
nStartChar	number	起始字符索引	否	无
nEndChar	number	终止字符索引	否	无

返回值

无

注意事项

nStartChar 零索引起始字符
中文字符占两个位置，比如”123 烈浪”中，烈字的索引起始为 3，终止为 4

示例

```
edit.SetSel(0,5)                    --取得字符串中前 5 个字符
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

Copy

拷贝文本框内容

```
Copy()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
edit.Copy()
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua
```

Paste

粘贴文本框内容

```
Paste()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
edit.Paste()
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua
```

Cut

剪切文本框内容

```
Cut()
```

参数

无

返回值

无

注意事项

无

示例

```
edit.Cut()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

Clear

清除文本框内容

```
Clear()
```

参数

无

返回值

无

注意事项

无

示例

```
edit.Clear()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

InsertText

在指定位置插入文本

InsertText(str, pos)

参数

参数名	类型	描述	是否可选	默认值
str	string	插入字符串	否	无
pos	number	插入位置	是	-1

返回值

无

注意事项

pos == -1 表示在末尾插入，索引以 0 开始

示例

edit.InsertText(“123”, 1)

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXEdit\脚本\XMain.lua

LXSlider

功能描述

该对象实现滑块控件功能。

函数列表

函数名称	说明
GetTotalSize	获取滑动最大值
SetTotalSize	设置滑动最大值
GetCurPos	获取滑块当前位置
SetCurPos	设置滑块当前位置
SetMargin	设置中线边界距

GetMargin	获取中线边界距

GetTotalSize

获取滑块最大值

```
GetTotalSize()
```

参数

无

返回值

类型	描述
number	滑块最大值

注意事项

无

示例

```
nRange = slider.GetTotalSize()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXSlider\脚本\XMain.lua

SetTotalSize

设置滑块最大值

```
SetTotalSize(nRange)
```

参数

参数名	类型	描述	是否可选	默认值
nRange	number	滑块最大值	无	无

返回值

无

注意事项

无

示例

```
slider.SetTotalSize(100)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXSlider\脚本\XMain.lua

GetCurPos

获取滑块当前位置

```
GetCurPos()
```

参数

无

返回值

类型	描述
number	滑块当前位置

注意事项

无

示例

```
nCurPos = slider.GetCurPos()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXSlider\脚本\XMain.lua

SetCurPos

设置滑块当前位置

```
SetCurPos(nCurPos)
```

参数

参数名	类型	描述	是否可选	默认值

nCurPos	number	滑块当前位置	无	无

返回值

无

注意事项

无

示例

```
slider.SetCurPos(50)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXSlider\脚本\XMain.lua

SetMargin

设置中线边界距

```
SetMargin(left, top, right, bottom)
```

参数

参数名	类型	描述	是否可选	默认值
left	number	滑块矩形区域左边的边距	否	无
top	number	滑块矩形区域上边的边距	否	无
right	number	滑块矩形区域右边的边距	否	无
bottom	number	滑块矩形区域下边的边距	否	无

返回值

无

注意事项

请注意，滑块的位置处在中线矩形的中央，例如当水平风格时，则滑块的 top 距离在中线矩形的中央，当为垂直风格时，滑块的 left 距离也在中线矩形的中央。

示例

```
slider.SetMargin(2, 2, 3, 3)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXSlider\脚本\XMain.lua

GetMargin

获取中线边界距

GetMargin()

参数

无

返回值

返回值	类型	描述
left	number	滑块矩形区域左边的边距
top	number	滑块矩形区域上边的边距
right	number	滑块矩形区域右边的边距
bottom	number	滑块矩形区域下边的边距

注意事项

无

示例

```
left, top, right, bottom = slider.GetMargin()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXSlider\脚本\XMain.lua

LXProgress

功能描述

该对象实现进度条功能。

函数列表

函数名称	说明
GetTotalRange	获取进度条范围
SetTotalRange	设置进度条范围
GetCurPos	获取进度条当前位置
SetCurPos	设置进度条当前位置
SetMargin	设置进度条边距
GetMargin	获取进度条边距

GetTotalRange

获取进度条范围

```
GetTotalRange()
```

参数

无

返回值

类型	描述
number	进度条范围

注意事项

无

示例

```
Range = Progress.GetTotalRange()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXProgress\脚本\XMain.lua

SetTotalRange

设置进度条范围

```
SetTotalRange(range)
```

参数

参数名	类型	描述	是否可选	默认值
range	number	进度条范围	否	无

返回值

无

注意事项

无

示例

```
Progress.SetTotalRange(100)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXProgress\脚本\XMain.lua

GetCurPos

获取当前位置

```
GetCurPos()
```

参数

无

返回值

类型	描述
number	进度条当前位置

注意事项

无

示例

```
Pos = Progress.GetCurPos()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXProgress\脚本\XMain.lua

SetCurPos

设置当前位置

```
SetCurPos(pos)
```

参数

参数名	类型	描述	是否可选	默认值
pos	number	设置进度条当前位置	否	无

返回值

无

注意事项

无

示例

```
Progress.SetCurPos(40)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXProgress\脚本\XMain.lua

SetMargin

设置进度条边界距

```
SetMargin(left, top, right, bottom)
```

参数

参数名	类型	描述	是否可选	默认值
left	number	进度条左边的边距	否	无
top	number	进度条上边的边距	否	无
right	number	进度条右边的边距	否	无
bottom	number	进度条下边的边距	否	无

返回值

无

注意事项

无

示例

```
Progress.SetMargin(10, 8, 35, 7)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXProgress\脚本\XMain.lua

GetMargin

获取进度条边界距

```
GetMargin()
```

参数

```
无
```

返回值

返回值	类型	描述
left	number	进度条左边的边距
top	number	进度条上边的边距
right	number	进度条右边的边距
bottom	number	进度条下边的边距

注意事项

```
无
```

示例

```
left, top, right, bottom = Progress.GetMargin()
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXProgress\脚本\XMain.lua
```

LXListCtrl

功能描述

实现对 ListCtrl 控件的使用

函数列表

函数名称	说明
DeleteAllItems	删除所有项
InsertItem	添加一项
InsertColumn	插入一列
DeleteColumn	删除一列
SetItemText	设置 Item 的文本

GetItemText	获取 Item 的文本
DeleteItem	删除一行
GetItemCount	得到行数
GetColCount	得到列数
SetItemData	设置每行数据
GetItemData	得到每行数据
GetItem	得到一项
SetItem	设置一项
GetItemState	得到项的状态
SetItemState	设置项的状态
GetFirstSelectItem	获取选中的项　返回选中 Item 中的第一项的位置
GetNextSelectItem	获取下一个被选中的 Item
GetImageList	获取图像链表
SetImageList	设置图像链表
GetSelectedColumn	获取选择列
SetSelectedColumn	设置选择列
GetHeadCtrl	获取 HeadCtrl 对象
HitTest	确定项的位置
SortItems	选项排序
SetItemInterval	设置 IconView 风格下项之间的间距
GetItemInterval	获取 IconView 风格下项之间的间距
SetIconSize	设置 Icon 的尺寸
GetIconSize	获取 Icon 的尺寸
SetItemImage	设置 Item 对应的图像索引
GetItemImage	获得 Item 对应的图像索引

DeleteAllItems

删除所有项

```
DeleteAllItems()
```

参数

无

返回值

无

注意事项

无

示例

```
listctrl.DeleteAllItems()
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua
```

InsertItem

添加一项

```
InsertItem(nItem, nSubItem, pszValue)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	行位置	否	无
nSubItem	number	列位置	否	无
Text	string	Item 的文本内容	否	无

返回值

类型	描述
boolean	0 失败 1 成功

注意事项

```
无
```

示例

```
listctrl.InsertItem(0, 0, "123")
listctrl.InsertItem(0, 0, "456")
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua
```

InsertColumn

插入一列

```
InsertColumn(nItem, nWidth，pszText, uType)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	列的序号	否	无
nWidth	number	列的宽度	否	无
pszText	string	列标题	否	无
uType	number	标题对齐方式	否	无

返回值

类型	描述
number	>0 添加成功

注意事项

uType 对齐方式
0 左对齐
1 居中对齐
2 右对齐

示例

```
listctrl.InsertColumn(0, 50, "1", 0)
listctrl.InsertColumn(0, 50, "2", 1)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

DeleteColumn

删除一列

```
DeleteColumn(nItem)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	列的序号	否	无

返回值

无

注意事项

无

示例

listctrl.DeleteColumn(0)

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

SetItemText

设置 Item 的文本

SetItemText(nItem，nSubItem，pszText)

参数

参数名	类型	描述	是否可选	默认值
nItem	number	逻辑行	否	无
nSubItem	number	逻辑列	否	无
pszText	string	文本内容	否	无

返回值

无

注意事项

无

示例

listctrl.SetItemText(0, 1, "kcl")

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetItemText

获取 Item 的文本

GetItemText(nItem， nSubItem)

参数

参数名	类型	描述	是否可选	默认值
nItem	number	逻辑行	否	无
nSubItem	number	逻辑列	否	无

返回值

类型	描述
string	文本内容

注意事项

无

示例

```
listctrl.GetItemText(0, 0)
listctrl.GetItemText(0, 1)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

DeleteItem

删除一行

```
DeleteItem(nItem)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	逻辑行	否	无

返回值

类型	描述
boolean	0 失败　1 成功

注意事项

无

示例

```
listctrl.DeleteItem(0)
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua
```

GetItemCount

得到行数

```
GetItemCount()
```

参数

```
无
```

返回值

类型	描述
number	Item 的个数

注意事项

```
无
```

示例

```
Count = listctrl.GetItemCount()
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua
```

GetColCount

得到列数

```
GetColCount()
```

参数

```
无
```

返回值

类型	描述

number	列的个数

注意事项

无

示例

```
Count = listctrl.GetColCount()
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

SetItemText

设置子 item 文本

```
SetItemText (nItem, nSubItem, Text)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	逻辑行	否	无
nSubItem	number	逻辑列	否	无
Text	string	文本内容	否	无

返回值

类型	描述
boolean	0 失败 1 成功

注意事项

无

示例

```
listctrl.SetItemText(0, 1, "kcl")
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetItemText

得到每行附加数据

```
GetItemText(nItem, nSubItem)
```

参数

参数名	类型	描述	是否可选	默认值
nIten	number	逻辑行	否	无
nSubItem	number	逻辑列	否	无

返回值

类型	描述
string	对应 Item 的文本

注意事项

无

示例

```
Str = listctrl.GetItemText(0, 1)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetItemState

得到项的状态

```
GetItemState(nItem)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	逻辑行	否	无

返回值

类型	描述
number	见注意事项

注意事项

```
各种返回状态参考值如下
X_SELECTED      0x01        //被选中
X_FOCUS         0x02        //有焦点
X_CHECKED       0x04        //被 Checked
X_PUSHED        0x08        //被按下
X_HOVER         0x10        //热点
X_DISABLED      0x20        //被禁止
```

示例

```
listctrl.GetItemState(1)
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua
```

SetItemState

设置项的状态

```
SetItemState(nItem, nState)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	逻辑行	否	无
nState	number	状态代码	否	无

返回值

类型	描述
boolean	1 成功　0 失败

注意事项

```
无
```

示例

```
listctrl.SetItemState(1, 1)
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua
```

GetFirstSelectItem

获取选中的项　返回选中 Item 中的第一项的位置

```
GetFirstSelectItem()
```

参数

无

返回值

类型	描述
number	第一个被选中项的逻辑位置,为 nil 表示没有选中

注意事项

无

示例

```
index = listctrl.GetFirstSelectItem()
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetNextSelectItem

获取下一个被选中的 Item

```
GetNextSelectItem(nItem)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	上一个选中项的位置	否	无

返回值

类型	描述
number	下一个被选中项的位置,为 nil 表示到尾

注意事项

无

示例

```
index = listctrl.GetFirstSelectItem()
Index = listctrl.GetNextSelectItem(index)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetImageList

获取图像链表

```
GetImageList(Type)
```

参数

参数名	类型	描述	是否可选	默认值
Type	number	需要得到的图像链表类型	否	无

返回值

类型	描述
LXImageList	Imagelist 的 lua 导出类

注意事项

Type
0 为小图标链
1 为大图标链

示例

```
imagelist = listctrl.GetImageList(0)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

SetImageList

设置图像链表

```
SetImageList(Type, LXImageList, IsShared)
```

参数

参数名	类型	描述	是否可选	默认值
LXImageList	Lua 导出类 LXImageList	Imagelist 的 Lua 导出类	否	无
Type	number	需要设置的图像链表类型	否	无
IsShared	boolean	图标链表是否共享（通常情况下为 1）	否	无

返回值

```
无
```

注意事项

```
Type = 0  小图标
Type = 1  大图标
请和函数 SetItemImage 结合使用才有效
```

示例

```
ImageList = LXImageList.new(32,32,32,0,4)
Image1 = LXHandle.new("D:/X.bmp")
Image2 = LXHandle.new("D:/Y.bmp")
Image3 = LXHandle.new("D:/Z.bmp")
ImageList.Add(Image1)
ImageList.Add(Image2)
ImageList.Add(Image3)
listctrl.SetImageList(ImageList,0,false)
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua
```

GetSelectedColumn

获取选择列

GetSelectedColumn()

参数

无

返回值

类型	描述
number	列索引

注意事项

无

示例

```
Index = listctrl.GetSelectedColumn()
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

SetSelectedColumn

设置选择列

SetSelectedColumn()

参数

参数名	类型	描述	是否可选	默认值
Column	number	列索引	否	无

返回值

无

注意事项

无

示例

```
listctrl.SetSelectedColumn(0)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetHeadCtrl

获取 LXHeadCtrl 对象指针

```
GetHeadCtrl()
```

参数

无

返回值

类型	描述
LXHeadCtrl	LXHeadCtrl 对象

注意事项

无

示例

```
headctrl = listctrl.GetHeadCtrl()
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

HitTest

确定项的位置

```
HitTest(point)
HitTest(x, y)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	X 位置	否	无
y	number	Y 位置	否	无
point	LXPoint	Point 的 lua 导出类	否	无

返回值

类型	描述

number	Item 位置索引

注意事项

无

示例

```
Point = LXPoint(20, 20)
listctrl.HitTest(point)
listctrl.HitTest(20, 20)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

SetItemInterval

设置 IconView 风格下项之间的间距

```
SetItemInterval(nIntrvlItem)
```

参数

参数名	类型	描述	是否可选	默认值
nIntrvlItem	number	IconView 风格下项之间的间距	否	无

返回值

无

注意事项

无

示例

```
listctrl.SetItemInterval(30)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetItemInterval

获得 IconView 风格下项之间的间距

GetItemInterval()

参数

无

返回值

类型	描述
number	IconView 风格下项之间的间距

注意事项

无

示例

```
listctrl.GetItemInterval()
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

SetIconSize

设置 Icon 的尺寸

SetIconSize()

参数

参数名	类型	描述	是否可选	默认值
width	number	宽度	否	无
heigth	number	高度	否	无

返回值

无

注意事项

无

示例

```
listctrl.SetIconSize(20, 20)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetIconSize

获取 Icon 的尺寸

GetIconSize()

参数

无

返回值

类型	描述
number	宽度
number	高度

注意事项

无

示例

Width, height = listctrl.GetIconSize()

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

SetItemImage

设置 Item 对应的图像索引

SetItemImage (nItem, nSubItem, Index)

参数

参数名	类型	描述	是否可选	默认值
nItem	number	行的索引	否	无
nSubItem	number	列的索引	否	无
Index	number	图像在列表中的索引	否	无

返回值

无

注意事项

请和函数 SetImageList 结合使用才有效

示例

listctrl. SetItemImage (0, 0, 1)

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

GetItemImage

获得 Item 对应的图像索引

GetItemImage (nItem, nSubItem)

参数

参数名	类型	描述	是否可选	默认值
nItem	number	行的索引	否	无
nSubItem	number	列的索引	否	无

返回值

类型	描述
number	图像在列表中的索引

注意事项

无

示例

n = listctrl. GetItemImage (0, 0)

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXListCtrl\脚本\XMain.lua

LXHeadCtrl

功能描述

该对象实现 HeaderCtrl 的功能需求。

函数列表

函数名称	说明
InsertColumn	插入一列
DeleteColumn	删除一列
DeleteAllColumn	删除所有列
GetSelectColumn	得到选择列
HitTest	确定项的位置
GetColumnCount	得到列的总数
GetColumnText	得到列的文本
SetColumnText	设置列的文本
GetColumnState	得到列的对齐方式
SetColumnState	设置列的对齐方式
GetColumnWidth	得到列的宽度
SetColumnWidth	设置列的宽度
SetColumnBKImage	设置列背景
GetColumnBKImage	得到列背景
GetAlignment	得到列的对齐方式
SetAlignment	设置列的对齐方式

InsertColumn

插入一列

```
InsertColumn (nItem, nWidth, pszText, uType)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	列索引	否	无
nWidth	number	宽度	否	无
pszText	string	文本	是	无
uType	number	文本对齐方式 0：左对齐 1：居中 2：右对齐	否	0

返回值

类型	描述
number	>0 成功

注意事项

无

示例

```
headctrl.InsertColumn(0, 50, "ydsh")
headctrl.InsertColumn(0, 50, "wsj", 1)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

DeleteColumn

删除一列

```
DeleteColumn (nItem)
```

参数

参数名	类型	描述	是否可选	默认值
nItem	number	列索引	否	无

返回值

类型	描述
boolean	0 失败 1 成功

注意事项

无

示例

```
headctrl.DeleteColumn(0)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

DeleteAllColumn

删除所有列

DeleteAllColumn ()

参数

无

返回值

类型	描述
boolean	0 失败 1 成功

注意事项

无

示例

headctrl.DeleteAllColumn ()

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

GetSelectColumn

得到选择列

GetSelectColumn ()

参数

无

返回值

类型	描述
number	被选中列的索引

注意事项

无

示例

```
headctrl.GetSelectColumn()
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua
```

HitTest

确定项的位置

```
HitTest (lxPoint)
HitTest(x, y)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	x 坐标	否	无
y	number	y 坐标	否	无
lxPoint	LXPoint	LXPoint	否	无

返回值

类型	描述
number	列索引

注意事项

无

示例

```
Point = LXPoint.new(20, 20)
headctrl.HitTest(point)
headctrl.HitTest(20, 20)
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua
```

GetColumnCount

得到列的总数

```
GetColumnCount()
```

参数

```
无
```

返回值

类型	描述
number	列的总数

注意事项

```
无
```

示例

```
headctrl.GetColumnCount()
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua
```

GetColumnText

得到列文本

```
GetColumnText(Index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	列的索引	否	无

返回值

类型	描述
string	文本

注意事项

```
无
```

示例

```
Str = headctrl.GetColumnText(1)
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua
```

SetColumnText

设置列的文本

```
SetColumnText (Index， Text)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	列索引	否	无
Text	string	文本内容	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
headctrl.SetColumnText(1, “456”)
```

参见

```
Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua
```

GetColumnState

得到列的状态

```
GetColumnState (Index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	列的索引	否	无

返回值

类型	描述
number	-1： 错误 0： 正常 0x02：有焦点 0x08：被按下 0x10：热点 0x20：被禁止

注意事项

无

示例

```
headctrl.GetColumnState(1)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

SetColumnState

设置列的状态

```
SetColumnState (Index，nState)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	列索引	否	无
nState	number	列状态，参见GetColumnState	否	无

返回值

无

注意事项

无

示例

```
headctrl.SetColumnState(1, 2)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

GetColumn Width

得到列宽度

GetColumnWidth (Index)

参数

参数名	类型	描述	是否可选	默认值
Index	number	列的索引	否	无

返回值

类型	描述
number	正值表示成功，-1 表示错误。

注意事项

无

示例

```
Width = headctrl.GetColumnWidth(1)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

SetColumnWidth

设置列的宽度

SetColumnWidth (Index，nWidth)

参数

参数名	类型	描述	是否可选	默认值
Index	number	列索引	否	无
nWidth	number	列宽度	否	无

返回值

无

注意事项

无

示例

```
headctrl.SetColumnWidth(1, 100)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

SetColumnBKImage

设置列背景

```
SetColumnBKImage (Path， DrawStyle，Index)
```

参数

参数名	类型	描述	是否可选	默认值
Path	string	背景图的路径	否	无
DrawStyle	number	绘制风格	否	无
Index	number	列索引	否	无

返回值

类型	描述
boolean	0 失败

注意事项

DrawStyle:
= 0　拉伸
= 1　平铺
= 2　九分图

示例

```
headctrl.SetColumnBKImage("D:\\x.bmp", 0, 0)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

GetColumnBKImage

得到列背景

GetColumnBKImage (Index)

参数

参数名	类型	描述	是否可选	默认值
Index	number	列索引	否	无

返回值

类型	描述
LXHandle	背景图句柄
DrawStyle	背景图的绘制风格

注意事项

无

示例

```
handle, dwStyle = headctrl.GetColumnBKImage(0)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

GetAlignment

得到列文本对齐方式

GetAlignment (Index)

参数

参数名	类型	描述	是否可选	默认值
Index	number	列索引	否	无

返回值

类型	描述
number	列的对齐方式

注意事项

返回值:
= 0 左对齐
= 1 居中对齐
= 2 右对齐

示例

```
Align = headctrl.GetAlignment(0)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

SetAlignment

设置列的对齐方式

```
SetAlignment (Index，nType)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	列索引	否	无
nType	number	对齐方式，其值参见 InsertColumn 函数 nType 参数	否	无

返回值

无

注意事项

无

示例

```
headctrl.SetAlignment(0, 1)
```

参见

Samples\Test_Wnd\Test_ListCtrl\Test_LXHeadCtrll\脚本\XMain.lua

LXComboBox

功能描述

该对象实现 ComboBox 控件功能。

函数列表

函数名称	说明
Addstring	添加项
Insertstring	插入项
Deletestring	删除项
GetCount	得到项的数量
GetTopIndex	得到当前被选择的项
SetTopIndex	设置当前选择的项
SetItemHeight	设置项的高度
GetItemHeight	得到项的高度
GetCurSel	在控件是单选风格时，得到当前被选择的项
SetCurSel	在控件是单选风格时，设置当前被选择的项
ResetContent	清空下拉框所有项
SetItemEnable	设置项的状态
SetDroppedHeight	设置下拉框 ListBox 的高度
GetDroppedHeight	得到下拉框 ListBox 的高度
GetImageList	获取下拉框 ListBox 图像链表
SetImageList	设置下拉框 ListBox 图像链表
GetEditSel	得到被选择的文本的位置，包括起始位置和末尾位置
SetEditSel	设置被选择的文本的位置，包括起始位置和末尾位置
LimitText	设置文本框的输入数量限制
ShowDropDown	显示或隐藏下拉列表
Findstring	根据文本内容查找项
Selectstring	根据文本内容选择项
GetLBText	得到项的文本内容
SetItemSpace	设置项的间距

Addstring

添加项

```
Addstring (lpszNewItem, nIndexImage, IsSetColor, crText)
```

参数

参数名	类型	描述	是否可选	默认值
lpszNewItem	string	表示文本内容	否	无
nIndexImage	number	表示项在图像 CImageList 中的索引	是	-1
IsSetColor	boolean	是否要设置文本颜色	是	false
crText	LXColor	表示要设置的文本的颜色	是	0

返回值

类型	描述
number	当前添加的项的序号

事项

1. 若未设置图像列表，则 nIndexImage 设置无效。
2. crText 在 IsSetColor 为 true 时有效。

示例

```
nRes = ComboBox.Addstring("11111", 0, 0 , 0);
color = LXColor.new(255,100,0, 0)
nRes = ComboBox.Addstring("11111", 0, true , color);
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

Insertstring

添加项

```
Insertstring (nPosition ,lpszNewItem, nIndexImage, IsSetColor, crText)
```

参数

参数名	类型	描述	是否可选	默认值
nPosition	number	表示插入的序号,	否	无

		若为-1，队尾插入		
lpszNewItem	string	表示文本内容	否	无
nIndexImage	number	表示项在图像CImageList中的索引	是	-1
IsSetColor	boolean	是否要设置文本颜色	是	false
crText	LXColor	表示要设置的文本的颜色	是	0

返回值

类型	描述
number	当前添加的项的序号

注意事项

1. 若未设置图像列表，则 nIndexImage 设置无效。
2. crText 在 IsSetColor 为 true 时有效。

示例

```
nRes = ComboBox. Insertstring (-1,"11111", 0, 0 , 0);
color = LXColor.new(255,100,0,255)
nRes = ComboBox. Insertstring (1, "11111", 0,true , color);
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

Deletestring

删除某一项

```
Deletestring (nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number	表示剩余的项的数量，-1 表示失败

注意事项

项的序号从 0 开始

示例

```
n = ComboBox.Deletestring(0)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

GetCount

得到项的数量

```
GetCount ()
```

参数

无

返回值

类型	描述
number	数量值

注意事项

无

示例

```
nRes = ComboBox.GetCount()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

GetTopIndex

得到当前最顶端的项

```
GetTopIndex ()
```

参数

无

返回值

类型	描述
number	表示项的序号

注意事项

示例

```
nIndex = ComboBox.GetTopIndex ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetTopIndex

设置当前最顶端的项

```
SetTopIndex (nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number	1 表示成功，0 表示失败

注意事项

项的序号从 0 开始

示例

```
bSuc = ComboBox.SetTopIndex (1)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetItemHeight

设置项的高度

```
SetItemHeight(height)
```

参数

参数名	类型	描述	是否可选	默认值
height	number	表示项的高度	否	无

返回值

无

注意事项

示例

```
ComboBox.SetItemHeight (20)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

GetItemHeight

得到项的高度

```
GetItemHeight ()
```

参数

无

返回值

类型	描述
number	高度值

注意事项

示例

```
nHeight = ComboBox.GetItemHeight ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

GetCurSel

在控件是单选风格时，得到当前被选择的项

GetCurSel ()

参数

无

返回值

类型	描述
number	当前被选择的项的序号, -1 表示失败

注意事项

无

示例

n = ComboBox.GetCurSel()

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetCurSel

在控件是单选风格时，设置当前被选择的项

SetCurSel (nIndex)

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number	当前被选择的项的序号, -1 表示失败

注意事项

无

示例

```
nIndex= ComboBox.SetCurSel (3)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

ResetContent

清空下拉框所有的项

```
ResetContent ()
```

参数

无

返回值

无

注意事项

示例

```
ComboBox.ResetContent ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetItemEnable

设置某个项的状态

```
SetItemEnable (nIndex, bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无
bEnable	boolean	是否允许项选中	否	无

返回值

类型	描述
number	返回项的序号

注意事项

项的序号从零开始

示例

```
nRes = ComboBox.SetItemEnable(0,false)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetDroppedHeight

设置下拉框 ListBox 的高度

```
SetDroppedHeight (nHeight)
```

参数

参数名	类型	描述	是否可选	默认值
nHeight	number	表示高度值	否	无

返回值

类型	描述
number	0 表示失败, 1 表示成功

注意事项

无

示例

```
bSuc = ComboBox. SetDroppedHeight (80)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

GetDroppedHeight

得到下拉框 ListBox 的高度

GetDroppedHeight ()

参数

无

返回值

类型	描述
number	高度值

注意事项

无

示例

```
nHeight = ComboBox.GetDroppedHeight()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetImageList

设置图像链表

SetImageList (ImageList, bVal)

参数

参数名	类型	描述	是否可选	默认值
ImageList	LXImageList 类型	要设置的图像列表值	否	无
bVal	boolean	该图形列表是否共享	否	无

返回值

无

返回值

无

注意事项

无

示例

```
ImageList =    LXImageList.new(32, 32, 25, 1, 1)
Handle = LXHandle.new("..\\Image\\x.jpg")
ImageList.Add(Handle);
Handle2 = LXHandle.new("..\\Image\\y.jpg")
ImageList.Add(Handle2);
ComboBox.SetImageList(ImageList, true)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua
```

GetImageList

得到下拉框 ComboBox 图像链表

```
GetImageList ()
```

参数

```
无
```

返回值

类型	描述
LXImageList	得到的图像列表值

注意事项

```
无
```

示例

```
Imagelist = ComboBox. GetImageList ()
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua
```

GetEditSel

得到被选择的文本的位置，包括起始位置和末尾位置

```
GetCurSel ()
```

参数

```
无
```

返回值

参数名	类型	描述	是否可选	默认值
nBegin	number	起始位置	否	无
nEnd	number	末尾位置	否	无

注意事项

无

示例

```
nBegin, nEnd = ComboBox.GetCurSel()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetEditSel

设置被选择的文本的位置，包括起始位置和末尾位置

```
SetCurSel (nBegin, nEnd)
```

参数

参数名	类型	描述	是否可选	默认值
nBegin	number	起始位置	否	无
nEnd	number	末尾位置	否	无

返回值

无

注意事项

起始位置从 0 开始

示例

```
ComboBox.SetCurSel (0, 3)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

LimitText

设置文本框的输入数量限制

```
LimitText (nMaxChars)
```

参数

参数名	类型	描述	是否可选	默认值
nMaxChars	number	表示输入数量	否	无

返回值

无

注意事项

示例

```
ComboBox.LimitText (5)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

ShowDropDown

显示或隐藏下拉列表

```
ShowDropDown(bVal)
```

参数

参数名	类型	描述	是否可选	默认值
bVal	boolean	true 表示显示， false 表示隐藏	否	无

返回值

无

注意事项

示例

```
ComboBox. ShowDropDown (true)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua
```

Findstring

根据文本内容查找项

```
Findstring (nStartAfter, lpszstring)
```

参数

参数名	类型	描述	是否可选	默认值
nStartAfter	number	查找的开始位置，从 0 开始	否	无
lpszstring	string	要查找的项的文本内容	否	无

返回值

类型	描述
number	当前查找到项的序号, -1 表示没有找到

注意事项

示例

```
nIndex = comboBox.Findstring (2, “123”)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua
```

Selectstring

根据文本内容选择项

```
Selectstring (nStartAfter, lpszstring)
```

参数

参数名	类型	描述	是否可选	默认值

nStartAfter	number	查找的开始位置，从 0 开始	否	无
lpszstring	string	要选择的项的文本内容	否	无

返回值

类型	描述
number	当前选择的项的序号, -1 表示没有选中任何项

注意事项

示例

```
nIndex = ComboBox.Selectstring (2, “123”)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

GetLBText

得到某个项的文本内容

```
GetLBText (nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
string	文本内容

注意事项

项的序号从 0 开始

示例

```
str = ComboBox.GetLBText(1)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

SetItemSpace

设置项与左边边框的显示间隔

SetItemSpace (nIndex, nSpace)

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无
nSpace	number	表示间距的象素距离数，0 表示无间距	否	无

返回值

无

注意事项

无

示例

```
ComboBox.SetItemSpace(1, 1)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXComboBox\脚本\XMain.lua

LXListBox

功能描述

该对象实现 ListBox 控件功能。

函数列表

函数名称	说明
Addstring	添加项

Insertstring	插入项
Deletestring	删除项
ResetContent	清空所有项
SetItemEnable	设置项的状态
GetCount	得到项的数量
SetImageList	设置图像链表
GetImageList	得到图像链表
SetLightBKColor	设置项处于高亮时的背景颜色
GetLightBKColor	得到项处于高亮时的背景颜色
SetGrayTextColor	设置项 disabled 时的文本颜色
GetGrayTextColor	得到项 disabled 时的文本颜色
GetGrayBKColor	设置项 disabled 时的背景颜色
SetGrayBKColor	得到项 disabled 时的背景颜色
GetSelectTextColor	设置项被选中时的文本颜色
SetSelectTextColor	得到项被选中时的文本颜色
GetSelectBKColor	设置项被选中时的背景颜色
SetSelectBKColor	得到项被选中时的背景颜色
GetTopIndex	得到当前最顶端的项
SetTopIndex	设置当前最顶端的项
GetItemHeight	得到项的高度
GetCurSel	在控件是单选风格时，得到当前被选择的项
SetCurSel	在控件是单选风格时，设置当前被选择的项
GetSel	在控件是多选风格时，得到某个项的状态，是否被选择
SetSel	在控件是多选风格时，设置某个项的状态，是否被选择
GetText	得到项的文本内容

Addstring

添加项

```
Addstring (lpszNewItem, nIndexImage, IsSetColor, crText)
```

参数

参数名	类型	描述	是否可选	默认值
lpszNewItem	string	表示文本内容	否	无
nIndexImage	number	表示项在图像列表 LXImageList 中的索引，默认值表示项按照图像列表依次显示	是	-1
IsSetColor	boolean	是否要设置文本颜	是	false

		色		
crText	LXColor	表示要设置的文本的颜色	是	0

返回值

类型	描述
number	当前添加的项的序号

注意事项

1. 若 listbox 无图像，则 nIndexImage 设置无效。
2. crText 在 IsSetColor 为 true 时有效。

示例

```
nRes = ListBox.Addstring("11111", 0, 0 , 0);
clor = LXColor.new(255,100,0,255)
nRes = ListBox.Addstring("11111", 0, true , clor);
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

Insertstring

添加项

```
Insertstring (nPosition ,lpszNewItem, nIndexImage, IsSetColor, crText)
```

参数

参数名	类型	描述	是否可选	默认值
nPosition	number	表示插入的序号，若为-1，末尾插入。若大于项的总数，则不执行操作，返回-1	否	无
lpszNewItem	string	表示文本内容	否	无
nIndexImage	number	表示项在图像列表 LXImageList 中的索引默认值表示项按照图像列表依次显示	是	-1
IsSetColor	boolean	是否要设置文本颜	是	false

		色		
crText	LXColor	表示要设置的文本的颜色	是	0

返回值

类型	描述
number	当前添加的项的序号，-1 表示失败

注意事项

1. 若 listbox 无图像，则 nIndexImage 设置无效。
2. crText 在 IsSetColor 为 true 时有效。

示例

```
nRes = ListBox. Insertstring (-1,"11111", 0, 0 , 0);
clor = LXColor.new(255,100,0,255)
nRes = ListBox.Insertstring (1, "11111", 0, true , clor);
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

Deletestring

删除某一项

```
Deletestring (nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number	表示剩余的项的数量，-1 表示失败

注意事项

项的序号从 0 开始

示例

```
n = ListBox.Deletestring(0)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

ResetContent

清空所有的项

```
ResetContent()
```

参数

```
无
```

返回值

```
无
```

注意事项

```

```

示例

```
Len = LisBox.ResetContent ()
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

SetItemEnable

设置某个项的状态

```
SetItemDisable (nIndex, bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无
bEnable	boolean	是否允许选择	否	无

返回值

类型	描述
number	返回项的序号

注意事项

项的序号从零开始
bEable
false 表示不允许
true 表示允许

示例

```
nRes = LisBox.SetItemEnable (0, true)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetCount

得到项的数量

```
GetCount()
```

参数

无

返回值

类型	描述
number	数量值

注意事项

无

示例

```
nRes = LisBox.GetCount()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

SetImageList

设置图像链表

```
SetImageList (ImageList, bVal)
```

参数

参数名	类型	描述	是否可选	默认值
ImageList	LXImageList	要设置的图像列表值	否	无
bVal	boolean	该图形列表是否共享	否	无

返回值

无

注意事项

无

示例

```
ImageList =    LXImageList.new(32, 32, 25, 1, 1)
Handle = LXHandle.new("..\\Image\\x.jpg")
ImageList.Add(Handle);
Handle2 = LXHandle.new("..\\Image\\x.jpg")
ImageList.Add(Handle2);
Handle3 = LXHandle.new("..\\Image\\y.jpg")
ImageList.Add(Handle3);
nRes = ListBox.SetImageList(ImageList,false);
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetImageList

得到图像链表

```
GetImageList ()
```

参数

无

返回值

类型	描述
LXImageList	得到的图像列表

注意事项

无

示例

```
Imagelist = ListBox.GetImageList()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

SetLightBKColor

设置项处于高亮时的背景颜色

```
SetLightBKColor (color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

示例

```
color = LXColor.new(255,100,0,0) --RGBA
ListBox.SetLightBKColor (color)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetLightBKColor

得到项处于高亮时的背景颜色

```
GetLightBKColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = ListBox.GetLightBKColor()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

SetGrayTextColor

设置项 disabled 时的文本颜色

```
SetGrayTextColor (color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

示例

```
color = LXColor.new(192,192,192,0) --RGBA
ListBox.SetGrayTextColor (color)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetGrayTextColor

得到项 disabled 时的文本颜色

```
GetGrayTextColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = ListBox. GetGrayTextColor ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

SetGrayBKColor

设置项 disabled 时的背景颜色

```
SetGrayBKColor (color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

示例

```
clor = LXColor.new(255,100,0,0) --RGBA
```

```
ListBox.SetGrayBKColor (clor)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

GetGrayBKColor

得到项 disabled 时的背景颜色

```
GetGrayBKColor ()
```

参数

```
无
```

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = ListBox.GetGrayBKColor ()
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

SetSelectTextColor

设置项处于高亮时的背景颜色

```
SetSelectTextColor (color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

```
无
```

注意事项

示例

```
color = LXColor.new(255,100,0,255) --RGBA
ListBox.SetSelectTextColor (color)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

GetSelectTextColor

得到项被选中时的文本颜色

```
GetSelectTextColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = ListBox.GetSelectTextColor ()
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

SetSelectBKColor

设置项被选中时的背景颜色

```
SetSelectBKColor (color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

示例

```
color = LXColor.new(255,100,0,0) --RGBA
ListBox.SetSelectBKColor (color)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetSelectBKColor

得到项被选中时的背景颜色

```
GetSelectBKColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = ListBox.GetSelectBKColor ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetTopIndex

得到当前最顶端的项

```
GetTopIndex ()
```

参数

无

返回值

类型	描述
number	表示项的序号

注意事项

示例

```
nIndex = ListBox.GetTopIndex ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

SetTopIndex

设置当前最顶端的项

```
SetTopIndex (nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number	1 表示成功，0 表示失败

注意事项

项的序号从 0 开始

示例

```
bSuc = ListBox. SetTopIndex (1)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetItemHeight

得到项的高度

GetItemHeight()

参数

无

返回值

类型	描述
number	高度值

注意事项

示例

nHeight = ListBox.GetItemHeight ()

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetCurSel

在控件是单选风格时，得到当前被选择的项

GetCurSel()

参数

无

返回值

类型	描述
number	返回当前被选择的项的序号, -1 表示失败

注意事项

无

示例

```
nIndex = ListBox.GetCurSel()
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

SetCurSel

在控件是单选风格时，设置当前被选择的项

```
SetCurSel (nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number 类型	当前被选择的项的序号, -1 表示失败

注意事项

```
无
```

示例

```
nIndex = ListBox.SetCurSel (3)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua
```

GetSel

在控件是多选风格时，得到某个项的状态，是否被选择

```
GetSel(nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number	-1 表示失败, 0 表示未选中, 1 表示选中

注意事项

项的序号从 0 开始

示例

```
n = ListBox.GetSel(1)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

SetSel

在控件是多选风格时，设置某个项的状态，是否被选择

```
SetCurSel (nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示项的序号	否	无

返回值

类型	描述
number	1 表示成功，-1 表示失败

注意事项

项的序号从 0 开始

示例

```
nIndex = ListBox.SetSel (1)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

GetText

得到项的文本内容

```
GetText(nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	表示要设置的项的序号	否	无

返回值

类型	描述
string	文本内容

注意事项

项的序号从 0 开始

示例

```
str = ListBox.GetText(1)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXListBox\脚本\XMain.lua

LXMenu

功能描述

该对象实现弹出菜单功能。

函数列表

函数名称	说明
AppendMenu	添加项
InsertMenu	插入项
TrackPopupMenu	弹出菜单
DeleteMenu	删除某一项
SetImageList	设置图像链表
GetImageList	得到图像链表
SetLightBKColor	设置项处于高亮时的背景颜色
GetLightBKColor	得到项处于高亮时的背景颜色
SetDisTextColor	设置项 disabled 时的文本颜色

GetDisTextColor	得到项 disabled 时的文本颜色
SetLeftPanelColor	设置竖线左侧边框的背景颜色
GetLeftPanelColor	得到竖线左侧边框的背景颜色
EnableMenuItem	设置某个项的状态
EnableMenuAllItem	设置所有项的状态

AppendMenu

添加项

```
AppendMenu (nFlags, pChildMenu, lpszNewItem, nID, chAltKey, nIndexImage, IsSetColor, crText)
```

参数

参数名	类型	描述	是否可选	默认值
nFlags	number	表示项的类型, 0 表示菜单文本, 16 表示子菜单 ,2048 表示分割横线	否	无
pChildMenu	LXMenu	子菜单，当 nFlags 为 16 时，该项有效	否	无
lpszNewItem	string	表示菜单项文本内容	否	无
nID	number	表示项的命令 ID, 如 123, 124	是	-1
chAltKey	string	表示快捷键	是	0
nIndexImage	number	表示项的图像在 CImageList 中的索引，若为默认参数，则按照 CImageList 的顺序依次显示	是	-1
IsSetColor	number	是否要设置文本颜色	是	false
crText	LXColor	表示要设置的文本的颜色	是	0

返回值

类型	描述
number	当前添加的项的序号

注意事项

nID, chAltKey, nIndexImage, IsSetColor, crText 可设默认值。
crText 在 IsSetColor 为 true 时有效。

示例

```
color = LXColor.new(255,100,0,255) --RGBA
n = Menu.AppendMenu(0, 0, "剪切", 0, "K");
n = Menu.AppendMenu(0, 0, "复制", 0, "C", 1 true, color);
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

InsertMenu

添加项

InsertMenu (nPosition,nFlags, pChildMenu, lpszNewItem, nID, chAltKey, nIndexImage, IsSetColor, crText)

参数

参数名	类型	描述	是否可选	默认值
nPosition	number	表示插入的序号，若为-1，在菜单末尾插入	否	无
nFlags	number	表示项的类型, 0 表示菜单文本, 16 表示子菜单, 2048 表示分割横线	否	无
pChildMenu	LXMenu	子菜单，当 nFlags 为 16 时，该项有效	否	无
lpszNewItem	string	表示菜单项文本内容	否	无
nID	number	表示项的 ID, 如 123, 124	是	-1
chAltKey	string	表示快捷键	是	0
nIndexImage	number	表示菜单项的图像在 CImageList 中的索引，若为默认参数，则按照 CImageList 的顺序依次显示	是	-1
IsSetColor	boolean	是否要设置文本颜色	是	false
crText	LXColor	表示要设置的文本	是	0

		的颜色		

返回值

类型	描述
number	当前插入的项的序号

注意事项

1.nID, chAltKey, nIndexImage, IsSetColor, crText 可设默认值。
2.crText 在 IsSetColor 为 true 时有效。

示例

```
color = LXColor.new(255,100,0,255)    --RGBA
n = Menu.InsertMenu(0, 0, 0, "粘贴", 0, "V", 1 ,0, 0);
n = Menu.InsertMenu(1, 0, 0, "全选", 0, "A", 2 ,1, color);
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

TrackPopupMenu

弹出菜单

```
TrackPopupMenu (nFlags, x, y, pWnd, lpRect)
```

参数

参数名	类型	描述	是否可选	默认值
nFlags	number	保留用	是	0
x	number	虚拟屏幕坐标的 left 值	否	无
y	number	虚拟屏幕坐标的 top 值	否	无
pWnd	LXWnd	接受菜单单击消息的窗口	否	无
lpRect	LXRect	这个值没有作用	是	nil

返回值

类型	描述
number	是否正确，1 表示正确，0 表示失败

注意事项

若该函数不调用，则菜单弹出位置既是控件创建的位置

示例

```
n = Menu. TrackPopupMenu (0, 0, 0, myDialog, 0)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

DeleteMenu

删除某一项

```
DeleteMenu (nID, nFlags)
```

参数

参数名	类型	描述	是否可选	默认值
nID	number	表示要删除的菜单项，意义依赖于 nFlags	否	无
nFlags	number	表示 nID 的类别，当为 0 时，表示 nID 是一个命令值，这个命令值就是 AppendMenu 函数的第四个参数，或者 InsertMenu 的第五个参数；当为 1024 时表示 nID 是项的序号，该序号从 0 开始	否	无

返回值

类型	描述
number	1 表示成功，0 表示失败

注意事项

nID 表示序号时，请注意项的序号是动态变化的，比如先插入三项，而后删除了一项，那么现在总共就只有两项，其序号依次为 0,1. 即使项的类型是分割横线，

该项的序号同样计算到菜单总数中。所以假使某个项是分隔横线，若该项序号是 1，则函数“n = Menu.DeleteMenu (1, 1024)”将删除该分割横线

示例

```
bSuc = Menu.DeleteMenu(1, 1024)
bSuc = Menu.DeleteMenu(20, 0)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

SetImageList

设置图像链表

```
SetImageList (ImageList, bVal)
```

参数

参数名	类型	描述	是否可选	默认值
ImageList	LXImageList 类型	要设置的图像列表	否	无
bVal	boolean	该图形列表是否共享	否	无

返回值

无

注意事项

无

示例

```
ImageList = LXImageList.new(32, 32, 25, 1, 1)
Handle = LXHandle.new( “..\\Image\\x.jpg”)
ImageList.Add(Handle);
Handle2 = LXHandle.new(“..\\Image\\y.jpg”)
ImageList.Add(Handle2);
Menu.SetImageList (ImageList, false)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

GetImageList

得到图像链表

```
GetImageList ()
```

参数

无

返回值

类型	描述
LXImageList	得到菜单的图像列表

注意事项

无

示例

```
Imagelist = Menu.GetImageList()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

SetLightBKColor

设置项处于高亮时的背景颜色

```
SetLightBKColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

示例

```
color = LXColor.new(255,100,0,255) --RGBA
Menu.SetLightBKColor (color)
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua
```

GetLightBKColor

得到项处于高亮时的背景颜色

```
GetLightBKColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = Menu.GetLightBKColor()
```

参见

```
Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua
```

SetDisTextColor

设置项 disabled 时的文本颜色

```
SetDisTextColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

示例

```
color = LXColor.new(255,100,0,255) --RGBA
Menu.SetDisTextColor (color)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

GetDisTextColor

得到项 disabled 时的文本颜色

```
GetDisTextColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = Menu.GetDisTextColor ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

SetLeftPanelColor

设置竖线左侧边框的背景颜色

```
SetLeftPanelColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

仅在菜单有“竖的分割线”的时候有用

示例

```
color = LXColor.new(255,100,0,255) --RGBA
Menu.SetLeftPanelColor (color)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

GetLeftPanelColor

得到竖线左侧边框的背景颜色

```
GetLeftPanelColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

示例

```
color = Menu.GetLeftPanelColor ()
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

EnableMenuItem

设置某个项的状态

```
EnableMenuItem (nID, bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
nID	number	表示要设置的项的序号	否	无
bEnable	boolean	表示状态值, true 是设置为活动的，false 是设置为不活动的	否	无

返回值

类型	描述
number	返回值应该与 nID 相同，否则返回- 1 表示失败

注意事项

项的序号从零开始

示例

```
nRes = Menu.EnableMenuItem (1, false)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

EnableMenuAllItem

设置所有项的状态

```
EnableMenuAllItem (bEnable)
```

参数

参数名	类型	描述	是否可选	默认值

bEnable	boolean	表示状态值,true 是设置为活动的，false 是设置为不活动的	否	无

返回值

无

注意事项

示例

```
Len = Menu.EnableMenuAllItem (false)
```

参见

Samples\Test_Wnd\Test_ComboBox\Test_LXMenu\脚本\XMain.lua

LXGroupBox

功能描述

该对象实现 GroupBox 控件功能。

函数列表

函数名称	说明
SetLineWidth	设置线宽
GetLineWidth	获取线宽
SetLineColor	设置线颜色
GetLineColor	获取线颜色

SetLineWidth

设置线宽

```
SetLineWidth(nWidth)
```

参数

参数名	类型	描述	是否可选	默认值
nWidth	number	线宽	否	无

返回值

无

注意事项

无

示例

```
GroupBox.SetLineWidth (3)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXGroupBox \脚本\XMain.lua

GetLineWidth

获取线宽

```
GetLineWidth()
```

参数

无

返回值

类型	描述
number	线宽

注意事项

无

示例

```
nWidth = GroupBox.GetLineWidth ()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXGroupBox \脚本\XMain.lua

SetLineColor

设置线颜色

```
SetLineColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
nWidth	number	线宽	否	无

返回值

无

注意事项

无

示例

```
color = LXColor.new(255, 255, 0, 255)
GroupBox.SetLineColor (color)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXGroupBox \脚本\XMain.lua

GetLineColor

获取线颜色

```
GetLineColor()
```

参数

无

返回值

类型	描述
LXColor	线的颜色值

注意事项

无

示例

```
color = GroupBox.GetLineColor()
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXGroupBox \脚本\XMain.lua
```

LXTabCtrl

功能描述

该对象实现 LXTabCtrl 控件功能。

函数列表

函数名称	说明
AddTabItem	添加一个 Tab 项
DeleteTabItem	删除一个 Tab 项
SetTabItemTip	设置 Tab 项提示信息
SetTabItemWnd	设置 Tab 项窗口
GetFirstTabItem	获取显示第一项的索引
SetFirstTabItem	设置显示第一项
GetTabItemImage	获取 Tab 项背景图
SetTabItemImage	设置 Tab 项背景图
GetScrollBtnImage	获取滚动按钮背景图
SetScrollBtnImage	设置滚动按钮背景图

AddTabItem

添加一个 Tab 项

```
AddTabItem(pGraphWnd, strInfo, strTip)
```

参数

参数名	类型	描述	是否可选	默认值
pGraphWnd	LXWnd	Tab 关联的窗口	否	无
strInfo	string	窗口说明	否	无
strTip	string	提示信息	否	无

返回值

无

注意事项

无

示例

```
testbutton = LXButton.new()
testbutton.CreateWindow("Button", "cv",0,0,400,400,tablectrl, 10009)
tablectrl.AddTabItem(testbutton, "testbutton", "testbutton")
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua

DeleteTabItem

添加一个 Tab 项

```
DeleteTabItem (index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	Tab 页的索引	否	无

返回值

无

注意事项

无

示例

```
Tablectrl.DeleteTabItem(0)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua

SetTabItemTip

设置 Tab 项提示信息

```
SetTabItemTip(nIdex, str)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	Tab 项索引	否	无
strTip	string	提示信息内容	否	无

返回值

无

注意事项

无

示例

```
Tablectrl.SetTabItemTip(0, “123”)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua

SetTabItemWnd

设置 Tab 项窗口

```
SetTabItemWnd(nIndex, pGraphWnd)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	Tab 项索引	否	无
pGraphWnd	LXWnd	Tab 项关联的窗口	否	无

返回值

无

注意事项

无

示例

```
testbutton = LXButton.new()
testbutton.CreateWindow("button", "cv",0,0,400,400,tablectrl, 10009)
tablectrl.AddTabItem(testbutton, "testbutton", "testbutton")
testStatic = LXStatic.new()
testStatic.CreateWindow("Static", "cv", 0,0,400,400,tablectrl, 10010)
tablectrl.SetTabItemWnd(0, testStatic)
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua
```

GetFirstTabItem

获取显示第一项的索引

```
GetFirstTabItem()
```

参数

```
无
```

返回值

类型	描述
number	显示第一项索引

注意事项

```
无
```

示例

```
nIndex = tablectrl.GetFirstTabItem()
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua
```

SetFirstTabItem

设置显示第一项

```
SetFirstTabItem(nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	Tab 项索引	否	无

返回值

无

注意事项

无

示例

```
Tablectrl.SetFirstTabItem(1)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua

GetTabItemImage

获取 Tab 项背景图

```
GetTabItemImage(Index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	对应的 Tab 的索引	否	无

返回值

类型	描述
LXHandle	Tab 项背景图片句柄
number	绘制方式

注意事项

无

示例

```
Handle, DrawType = tablectrl.GetTabItemImage(1)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua

SetTabItemImage

设置 Tab 项背景图

SetTabItemImage(strFileName, DrawType, Index)

参数

参数名	类型	描述	是否可选	默认值
strFileName	string	Tab 背景图片绝对路径	否	无
DrawType	number	绘制方式	是	0
Index	number	对应的 Tab 的索引	是	0

返回值

成功返回大于等于 0 失败返回小于 0

注意事项

绘制方式 DrawType 取值如下:
= 0 图像平铺 （默认值）
= 1 图像拉伸
= 2 九分图绘制

示例

```
local szCurPath = GetXsnPath()
Tablectrl.SetTabItemImage (szCurPath .. "Image\\x.jpg", 0, 1)
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua

GetScrollBtnImage

获取滑动按钮图片

GetScrollBtnImage()

参数

无

返回值

类型	描述
LXHandle	滑动按钮图片句柄
number	绘制方式

注意事项

无

示例

```
Handle, DrawType = tablectrl.GetScrollBtnImage()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua

SetScrollBtnImage

设置滑动按钮图片

```
SetScrollBtnImage(strFileName, DrawType)
```

参数

参数名	类型	描述	是否可选	默认值
strFileName	string	滑动按钮图片绝对路径	否	无
DrawType	number	绘制方式	是	0

返回值

成功返回大于等于 0 失败返回小于 0

注意事项

绘制方式 DrawType 取值如下:
= 0 图像拉伸 （默认值）
= 1 图像平铺
= 2 九分图绘制

示例

```
local szCurPath = GetXsnPath()
```

```
Tablectrl.SetScrollBtnImage(szCurPath .. "Image\\x.jpg", 1)
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXTabCtrl\脚本\XMain.lua
```

LXToolTip

功能描述

该对象实现 LXToolTip 功能。

函数列表

函数名称	说明
ShowTip	显示 ToolTip
HideTip	隐藏 ToolTip
SetTipText	设置 ToolTip 文字
SetDelayTime	设置延迟显示时间
SetDisplayTime	设置显示时间
GetDelayTime	获取延迟时间
GetDisplayTime	设置显示时间

ShowTip

显示 ToolTip

```
ShowTip()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
tooltip.ShowTip()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXToolTip\脚本\XMain.lua

HideTip

隐藏 ToolTip

HideTip()

参数

无

返回值

无

注意事项

无

示例

Tooltip.HideTip()

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXToolTip\脚本\XMain.lua

SetTipText

设置 ToolTip 文字

SetTipText(strText)

参数

参数名	类型	描述	是否可选	默认值
strText	string	显示的文本	否	无

返回值

无

注意事项

无

示例

```
Tooltip.SetTipText("欢迎你使用 tooltip")
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXToolTip\脚本\XMain.lua
```

SetDelayTime

设置从即将显示到开始显示经历的时间

```
SetDelayTime(seconds)
```

参数

参数名	类型	描述	是否可选	默认值
seconds	number	时间值	否	无

返回值

```
无
```

注意事项

```
seconds 单位为毫秒
```

示例

```
Tooltip.SetDelayTime (1000)
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXToolTip\脚本\XMain.lua
```

SetDisplayTime

设置真正显示所持续的时间

```
SetDisplayTime(seconds)
```

参数

参数名	类型	描述	是否可选	默认值
seconds	number	时间值	否	无

返回值

```
无
```

注意事项

```
seconds 单位为毫秒
```

示例

```
Tooltip.SetDisplayTime (3000)
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXToolTip\脚本\XMain.lua
```

GetDelayTime

获取从即将显示到开始显示经历的时间

```
GetDelayTime()
```

参数

```
无
```

返回值

类型	描述
number	时间值

注意事项

```
无
```

示例

```
Seconds =Tooltip.GetDelayTime ()
```

参见

```
Samples\Test_Wnd\Test_Ctrls\Test_LXToolTip\脚本\XMain.lua
```

GetDisplayTime

获取真正的持续时间

```
GetDisplayTime()
```

参数

```
无
```

返回值

类型	描述
number	时间值

注意事项

无

示例

```
seconds = Tooltip.GetDisplayTime ()
```

参见

Samples\Test_Wnd\Test_Ctrls\Test_LXToolTip\脚本\XMain.lua

LXTreeCtrl

功能描述

该对象实现树形控件功能。

函数列表

函数名称	说明
InsertItem	添加节点
GetNextItem	根据参数不同，得到节点的下一个节点
GetChildItem	得到节点的子节点
GetParentItem	得到节点的父节点
GetRootItem	得到树的根节点
GetLastVisibleItem	得到当前视图中最后一个可视的节点
GetSelectedItem	得到当前被选定的节点
GetDropHilightItem	得到当前被拖放操作的节点
GetNextSiblingItem	得到节点的下一个兄弟节点
GetPrevSiblingItem	得到节点的前一个兄弟节点
GetFirstVisibleItem	得到第一个可视的节点
GetNextVisibleItem	得到节点的下一个可视的节点
GetPrevVisibleItem	得到节点的前一个可视的节点
ItemHasChildren	当前节点是否有子节点
GetCheck	得到节点是否被选择
SetCheck	设置节点是否被选择
SelectSetFirstVisible	滚动进入视图以使节点成为第一个可视节点
SelectDropTarget	设置拖放操作的节点以进行刷新
SelectItem	选定某一个节点

Expand	展开\收起某个节点
DeleteAllItems	删除所有节点
DeleteItem	删除某一个节点
HitTest	得到鼠标当前位置对应的节点
SetItem	设置节点的属性
Select	选定、滚动进入视图或者刷新某个节点
GetItemText	得到节点的文本
GetItemImage	得到与节点相关联的图形
GetItemState	得到节点的状态
SetItemText	设置节点的文本
SetItemImage	设置与节点相关联的图形
SetItemState	设置节点的状态
GetImageList	得到图像链表
SetImageList	设置图像链表
SetItemHeight	设置节点的高度
GetItemHeight	得到节点的高度
GetIndent	得到父节点与子节点在水平上的间隔
SetIndent	设置父节点与子节点在水平上的间隔
GetCount	得到节点的数量
GetVisibleCount	得到可视的节点的数量
SetTextColor	设置节点的文本的颜色
GetTextColor	得到节点的文本的颜色
SetSelTextColor	设置节点被选择的文本的颜色
GetSelTextColor	得到节点被选择的文本的颜色
SetSelBkColor	设置节点被选择的文本的背景颜色
GetSelBkColor	得到节点被选择的文本的背景颜色

InsertItem

添加节点

```
InsertItem (strName, strState, hParent, hInsertAfter, nImage, lParam)
```

参数

参数名	类型	描述	是否可选	默认值
strName	string	表示节点的文本内容	否	无
strState	string	表示节点的状态	是	“e”
hParent	节点句柄	节点的父节点	是	nil
hInsertAfter	节点句柄	插入节点	是	nil
nImage	number	表示节点的图像，I	是	-1

		mageList 中的索引		
lParam	number	表示节点的附加数据	是	0

返回值

类型	描述
节点句柄	添加成功后生成的节点

注意事项

State 参数取值如下:
“e” 创建可展开节点
“c” 创建带 CheckBox 节点
“d” 创建可以进行拖动操作的节点
“s” 创建可选中节点

示例

```
hItem = TreeCtrl.InsertItem(“新建节点”)                    --添加一个节点
hChild1 = TreeCtrl.InsertItem(“新建子节点 1”, -1, hItem)    --添加一个子节点
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetNextItem

根据参数不同，得到节点的下一个节点

```
GetNextItem (hItem, Code)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无
Code	number	节点类型	否	无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

```
Code 取值如下
= 0        父亲节点的第一个孩子节点
= 1        下一个兄弟节点
= 2        上一个兄弟节点
= 3        父亲节点
= 4        孩子结点
= 5        第一个可见节点
= 6        下一个可见节点
= 7        前一个可见节点
= 8        拖拽节点
= 9        得到当前选择节点
```

示例

```
hItem = TreeCtrl.GetSelectedItem()
hChildItem = TreeCtrl.GetNextItem(hItem, 4)
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

GetChildItem

得到指定节点的子节点

```
GetChildItem (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	指定节点	否	无

返回值

类型	描述
节点句柄	指定节点的子节点

注意事项

```
无
```

示例

```
--判断当前选中节点的子节点
hItem = TreeCtrl.GetSelectedItem()
```

```
if hItem ~= nil then
    hChildItem = TreeCtrl.GetChildItem(hItem)
end
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

GetParentItem

得到节点的父节点

```
GetParentItem (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	指定节点	否	无

返回值

类型	描述
节点句柄	指定节点的父节点

注意事项

```
无
```

示例

```
--判断当前选中节点的父节点
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
    hParentItem = TreeCtrl.GetParentItem(hItem)
end
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

GetRootItem

得到虚拟根节点

GetRootItem ()

参数

无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

这是一个虚拟节点，在控件中并不会显示出来，如果将其作为插入节点的父节点则生成的节点与调用函数 InsertItem 时不指定父节点效果相同

示例

```
hRoot = TreeItem.GetRootItem()
--下面两个 InsertItem 函数实现的效果相同
TreeCtrl.InsertItem(“新建节点”, “ecd”, hRoot)
TreeCtrl.InsertItem(“新建节点”, “ecd”)
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetLastVisibleItem

得到当前视图中最后一个可视的节点

GetLastVisibleItem ()

参数

无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

参见 SelectItem

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetSelectedItem

得到当前被选定的节点

GetSelectedItem ()

参数

无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

参见 GetNextVisibleItem

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetDropHilightItem

得到当前被拖放操作的节点

GetDropHilightItem ()

参数

无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

```
hItem = TreeCtrl.GetDropHilightItem()
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetNextSiblingItem

得到节点的下一个兄弟节点

```
GetNextSiblingItem (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

```
--判断当前选中节点的下一个兄弟节点
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
    hSiblingItem = TreeCtrl.GetNextSiblingItem(hItem)
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetPrevSiblingItem

得到节点的前一个兄弟节点

GetPrevSiblingItem (hItem)

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

```
--判断当前选中节点的前一个兄弟节点
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
    hSiblingItem = TreeCtrl.GetPrevSiblingItem(hItem)
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetFirstVisibleItem

得到第一个可视的节点

GetFirstVisibleItem ()

参数

无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

参见 SetCheck

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetNextVisibleItem

得到节点的下一个可视的节点

```
GetNextVisibleItem(hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

参见 SetCheck

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetPrevVisibleItem

得到节点的前一个可视的节点

```
GetPrevVisibleItem (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
节点句柄	表示一个节点

注意事项

无

示例

```
--判断当前选中节点的前一个可视节点
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
    hPrevItem = TreeCtrl.GetPrevVisibleItem(hItem)
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

ItemHasChildren

当前节点是否有子节点

```
ItemHasChildren (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
boolean	1 表示肯定，0 表示否定

注意事项

无

示例

```
--判断当前选中节点是否有子节点
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
    bHas = TreeCtrl.ItemHasChildren(hItem)
    print(bHas)
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetCheck

得到节点是否被选择

```
GetCheck (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
boolean	1 表示肯定，0 表示否定

注意事项

无

示例

参见 SetCheck

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

SetCheck

设置节点是否被选择

```
SetCheck (hItem, fCheck)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无
fCheck	boolean	1 表示肯定，0 表示否定	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

```
无
```

示例

```
Count = TreeCtrl.GetVisibleCount()
hItem = TreeCtrl.GetFirstVisibleItem()
--遍历所有节点的 check state
for I = 1, Count, 1 then
if hItem ~= nil then
	TreeCtrl.SetCheck(hItem, TreeCtrl.GetCheck(hItem))
	hItem = TreeCtrl.GetNextVisibleItem(hItem)
end
end
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

SelectSetFirstVisible

选中一个节点，并将其设置为第一可视节点

```
SelectSetFirstVisible (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

```
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
	TreeCtrl.SelectSetFirstVisible(hItem)
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SelectDropTarget

设置拖放操作的节点以进行刷新

```
SelectDropTarget (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

```
--设置鼠标所位于得节点为拖放操作节点
function OnMouseMove(this, nEventID, SendName, param)
	Point = LXPoint.new(param.x, param.y)
	hItem, state = HitText(Point)
	TreeCtrl.SelectDropTarget(hItem)
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SelectItem

选定某一个节点

```
SelectItem (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

```
hItem = TreeCtrl.GetLastVisibleItem()
if hItem ~= nil then
	TreeCtrl.SelectItem(hItem)	--选中最后一个可视节点
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

Expand

展开\收起某一个节点

Expand (hItem, strCode)

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无
strCode	string	表示标志	是	“Collapse”

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

strCode 取值如下:
“Collapse” 合并节点.
“CollapseReset” 合并节点并且删除子节点
“Expand” 展开节点
“Toggle” 反操作即如果当前展开状态就合并，当前合并状态就展开

示例

TreeCtrl.Expand(hItem, “Expand”)

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

DeleteAllItems

删除所有节点

DeleteAllItems ()

参数

无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

参见 GetCount

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

DeleteItem

删除所有节点

```
DeleteItem (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

```
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
	TreeCtrl.DeleteItme(hItem)		--删除选中节点
end
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

HitTest

得到鼠标当前位置对应的节点

```
HitTest (pt)
```

参数

参数名	类型	描述	是否可选	默认值
pt	LXPoint 实例	坐标点值	否	无

返回值

类型	描述
节点句柄	表示一个节点
number	表示节点的标志位，既是鼠标在节点的哪个部分上 0:Expand. 1:Check. 2:Image. 3:Text <0:其它

注意事项

```
无
```

示例

```
--判断鼠标按下对应的节点
function OnLButtonDown(this, nEventID, SendName, param)
	Point = LXPoint.new(param.x, param.y)
	hItem, state = HitText(Point)
end
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

SetItem

设置节点的属性

```
SetItem (hItem, Text, Image, State, Param)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无
Text	string	节点的文本内容	否	无
Image	long	节点的图形	是	-1
State	string	节点的状态	是	“e”
Param	number	表示附加数据	是	0

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

State 取值如下:
“c” checkbox 被选中
“d” 拖动操作
“e” 展开
“s” 被选中

示例

无

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

Select

选定、滚动进入视图或者刷新某个节点

```
Select(hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

无

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetItemText

得到节点的文本

```
GetItemText (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
string	节点的文本串

注意事项

无

示例

```
hItem = TreeCtrl.GetSelectedItem()
if hItem ~= nil then
```

```
    TreeCtrl.SetItemText(hItem, “aaa”)
    Str = TreeCtrl.GetItemText(hItem)
    print(str)                --输出 aaa
end
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

GetItemImage

得到与节点相关联的图形

```
GetItemImage (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
number	指定节点图片在 ImageList 中的索引

注意事项

```
无
```

示例

```
hItem = TreeCtrl.GetSelectedItem()
if TreeCtrl.ItemHasChildren(hItem) then
while hItem ~= nil then
      nImage = TreeCtrl.GetItemImage(hItem)
      print(nImage)
end
end
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

GetItemState

得到节点的状态

```
GetItemState (hItem)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无

返回值

类型	描述
string	节点的状态

注意事项

无

示例

参见 SetItemState

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetItemText

设置节点的文本

```
SetItemText (hItem, str)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无
str	string	表示节点的文本	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

参见 GetItemText

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetItemImage

设置与节点相关联的图形

```
SetItemImage (hItem, nImage)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无
nImage	number	表示 CImageList 中的序号,注意是从 0 开始	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

无

示例

无

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetItemState

设置节点的状态

```
SetItemState (hItem, State)
```

参数

参数名	类型	描述	是否可选	默认值
hItem	节点句柄	表示一个节点	否	无
State	string	表示状态值	否	无

返回值

类型	描述
boolean	1 表示操作成功，0 表示操作失败

注意事项

State 取值如下:
“c” checkbox 被选中
“d” 拖动操作
“e” 展开
“s” 被选中

示例

```
TreeCtrl.SetItemState(hItem, “c”)
local State = TreeCtrl.GetItemState(hItem)
print(State)                    --输出”c”
```

参见

```
Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua
```

GetImageList

得到下拉框 TreeCtrl 图像链表

```
GetImageList ()
```

参数

```
无
```

返回值

类型	描述
LXImageList	得到的图像列表值

注意事项

无

示例

```
LXImageListVal = TreeCtrl. GetImageList ()
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetImageList

设置图像链表

```
SetImageList (ImageList, bVal)
```

参数

参数名	类型	描述	是否可选	默认值
ImageList	LXImageList 类型	要设置的图像列表值	否	无
bVal	boolean	该图形列表是否共享	是	true

返回值

无

注意事项

无

示例

```
ImageList =    LXImageList.new(32, 32, 25, 1, 1)
Handle = LXHandle.new("D:\\x.bmp")
ImageList.Add(Handle);
Handle2 = LXHandle.new("D:\\y.bmp")
ImageList.Add(Handle2);
Handle3 = LXHandle.new (" D:\\z.bmp")
ImageList.Add(Handle3);
n = TreeCtrl.SetImageList (ImageList, 0)
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetItemHeight

设置节点的高度

```
SetItemHeight (height)
```

参数

参数名	类型	描述	是否可选	默认值
height	number	表示高度	否	无

返回值

类型	描述
number	返回设置前节点的高度

注意事项

无

示例

参见 GetItemHeight

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetItemHeight

得到节点的高度

```
GetItemHeight ()
```

参数

无

返回值

类型	描述
number	高度值

注意事项

无

示例

```
Height = TreeCtrl.GetItemHeight()
TreeCtrl.SetItemHeight(2*Height)          --节点高度变为原来的 2 倍
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetIndent

得到父节点与子节点在水平上的间隔

```
GetIndent ()
```

参数

无

返回值

类型	描述
number	高度值

注意事项

无

示例

```
Indent = TreeCtrl.GetIndent()
TreeCtrl.SetIndent(2*Indent)          --间隔变为原来的 2 倍
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetIndent

设置父节点与子节点在水平上的间隔

```
SetIndent (nVal)
```

参数

参数名	类型	描述	是否可选	默认值
nVal	number	表示间隔	否	无

返回值

无

注意事项

无

示例

参见 GetIndent

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetCount

得到节点的数量

```
GetCount ()
```

参数

无

返回值

类型	描述
number	数量值

注意事项

无

示例

```
TreeCtrl.DeleteAllItems()      --删除所有节点
if TreeCtrl.GetCount() == 0 then
    print(“No Item”)
end
if TreeCtrl.GetVisibleCount() == 0 then
    print(“No Visible Item”)
end
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetVisibleCount

得到可视的节点的数量

```
GetVisibleCount ()
```

参数

无

返回值

类型	描述
number	数量值

注意事项

无

示例

参见 GetCount

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetTextColor

设置节点的文本的颜色

```
SetTextColor (clor)
```

参数

参数名	类型	描述	是否可选	默认值
clor	LXColor 实例	颜色值	否	无

返回值

无

注意事项

无

示例

```
Color = LXColor.new(255,0,0)
TreeCtrl.SetTextColor(Color)                    --设置被选择文本背景为红色
Color1 = TreeCtrl.GetTextColor()
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetTextColor

得到节点的文本的颜色

```
GetTextColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

无

示例

参见 SetTextColor

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetSelTextColor

设置节点被选择的文本的颜色

```
SetSelTextColor (clor)
```

参数

参数名	类型	描述	是否可选	默认值
clor	LXColor	颜色值	否	无

返回值

无

注意事项

无

示例

```
Color = LXColor.new(255,0,0)
TreeCtrl.SetSelTextColor(Color)                    --设置被选择文本颜色为红色
Color1 = TreeCtrl.GetSelTextColor()
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetSelTextColor

得到节点被选择的文本的颜色

```
GetSelTextColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

无

示例

参见 SetSelTextColor

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

SetSelBkColor

设置节点被选择的文本的背景颜色

```
SetSelBkColor (clor)
```

参数

参数名	类型	描述	是否可选	默认值
clor	LXColor	颜色值	否	无

返回值

无

注意事项

无

示例

```
Color = LXColor.new(255,0,0)
TreeCtrl.SetSelBkColor(Color)                    --设置被选择文本背景为红色
Color1 = TreeCtrl.GetSelBkColor()
```

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

GetSelBkColor

得到节点被选择的文本的背景颜色

```
GetSelBkColor ()
```

参数

无

返回值

类型	描述
LXColor	颜色值

注意事项

无

示例

参见 SetSelBkColor

参见

Samples\Test_Wnd\Test_LXTreeCtrl\脚本\XMain.lua

LXDialog

功能描述

实现对话框操作，直接需要注意的是，除以下函数外，LXWnd 的函数如 CreateWindow、ShowWindow、DestroyWindow 也经常用来操作非模态对话框，GetDlgItem 函数用来得到子控件，敬请参考。

函数列表

函数名称	说明
new	创建窗口对象
create	创建非模态窗口
DoModal	创建并显示模态窗口
GetDlgItem	通过控件 ID 得到对话框控件对象
EndDialog	销毁模态窗口

new

创建窗口对象

```
Dlg = LXDialog.new(str)
```

参数

参数名	类型	描述	是否可选	默认值
str	string	Xrc 文件名	是	nil

返回值

类型	描述

LXDialog　　创建成功返回一个有效的窗口对象

注意事项

无

示例

```
Dlg = LXDialog.new()
Dlg = LXDialog.new("1.xrc")
```

参见

Samples\Test_Wnd\Test_LXDialog\脚本\XMain.lua

create

创建非模态对话框

```
create(Wnd)
```

参数

参数名	类型	描述	是否可选	默认值
Wnd	LXWnd	窗口对象	是	nil

返回值

无

注意事项

1. CreateWindow 也可创建非模态对话框，所不同的是，若 new 函数的参数为某个 xrc 文件名时，则该对象直接调用 create 即可，否则需调用 CreateWindow 来指定窗口属性。
2. 该函数经常与 ShowWindow，DestroyWindow 一起使用，用来操作非模态窗口

示例

```
myDialog.create()
或 myDialog. CreateWindow ("TestButton", "pv", 0, 0, 200,20, nil, 3)
```

参见

Samples\Test_Wnd\Test_LXDialog\脚本\XMain.lua

DoModal

创建并显示模态对话框

```
DoModal()
```

参数

无

返回值

类型	描述
number	ID_OK 表示确定，ID_CANCEL 表示取消

注意事项

1. 销毁模态窗口时务必使用 EndDialog 函数，而不能使用 LXWnd 的 DestroyWindow 函数
2. 窗口在调用该函数前不能调用 create 函数。因为该函数本身创建了窗口。

示例

无

参见

Samples\Test_Wnd\Test_LXDialog\脚本\XMain.lua

EndDialog

销毁模式窗口

```
EndDialog(ID)
```

参数

参数名	类型	描述	是否可选	默认值
ID	number	销毁模式窗口的标志，该值既是 DoModal 的返回值	否	无

返回值

无

注意事项

无

示例

```
LXDialog. EndDialog ()
```

参见

Samples\Test_Wnd\Test_LXDialog\脚本\XMain.lua 的 OnStartRender 函数

LXColorDialog

功能描述

实现颜色对话框

函数列表

函数名称	说明
new	创建一个 LXColorDialg 对象
GetColor	获取当前颜色
SetCurrentColor	设置当前颜色
GetSavedCustomColors	获取自定义颜色列表

new

创建一个 LXColorDialog 对象 s

```
new(color, parent)
```

参数

参数名	类型	描述	是否可选	默认值
color	颜色 LXColor 对象	选择的默认颜色	是	白色
parent	窗口 LXWnd 对象	父窗口	是	nil

返回值

类型	描述
LXColorDialog	创建成功返回一个有效的 LXColorDialog 窗口对象 ,否则返回 nil

注意事项

```
local clrDlg = LXColorDialog.new()
clfDlg.DoModal()
```

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXColorDialog\脚本\XMain.lua

GetColor

获得当前选择的颜色

```
GetColor()
```

参数

无

返回值

类型	描述
LXColor	返回一个 LXColor 颜色对象

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXhColorDialog\脚本\XMain.lua

SetCurrentColor

设置当前颜色

```
SetCurrentColor(color)
```

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色值	否	无

返回值

无

注意事项

在 DoModal 之前调用

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXColorDialog\脚本\XMain.lua

GetSavedCustomColors

获取自定义颜色列表

```
GetSavedCustomColors()
```

参数

无

返回值

类型	描述
table	返回一个自定义颜色列表

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXColorDialog\脚本\XMain.lua

LXFileDialog

功能描述

实现字体对话

函数列表

函数名称	说明
new	创建一个 LXFileDialog 对象
GetFileExt	获取文件扩展名
GetFileName	获取文件名
GetFileTitle	获得输入的文件名的标题
GetPathName	获得输入文件的全路径
SetDefExt	设置得默认扩展名
GetFolderPath	获取打开文件夹的目录名
AddShortCutFolder	添加一个文件夹快捷按钮
GetAllPathNames	获取多个选择的文件或目录

new

创建一个 LXFileDialog 对象

```
new(bOpenFileDialog, strDefExt, strFileName, strFlags, strFilter, pParentWnd)
```

参数

参数名	类型	描述	是否可选	默认值
bOpenFileDialog	Boolean	打开还是保存	是	true
strDefExt	string	扩展名	是	nil
strFileName	string	初始时文件名框中出现的名称	是	nil
strFlags	string	标志	是	nil
strFilter	string	过滤字符串	是	nil
parentWnd	窗口对象	父窗口	是	nil

返回值

类型	描述
LXFileDialog	创建成功返回一个有效的 LXFileDialog 窗口对象 ,否则返回 nil

注意事项

strFlags 为以下的值
‘a’ 表示可以选择多个文件
‘p’ 表示选择的是目录

示例

```
fileDlg =LXFileDialog.new()
```

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

GetFileExt

获取文件扩展名

```
GetFileExt()
```

参数

无

返回值

类型	描述
string	获取文件扩展名

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

GetFileName

获取文件名

```
GetFileName()
```

参数

无

返回值

类型	描述
string	获取文件名

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

GetFileTitle

获取文件标题

```
GetFileTitle()
```

参数

无

返回值

类型	描述
string	获取文件标题

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

GetPathName

获取输入文件的全路径

GetPathName()

参数

无

返回值

类型	描述
string	获取输入文件的全路径

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

SetDefExt

设置得默认扩展名

SetDefExt(strDefExt)

参数

参数名	类型	描述	是否可选	默认值
strDefExt	string	默认扩展名	否	无

返回值

无

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

GetFolderPath

获取打开文件夹的目录名

GetFolderPath ()

参数

无

返回值

类型	描述
string	获取打开文件夹的目录名

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

AddShortCutFolder

设置得默认扩展名

AddShortCutFolder (strFold)

参数

参数名	类型	描述	是否可选	默认值
strFold	string	路径	是	nil

返回值

无

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

GetAllPathNames

GetAllPathNames()

参数

无

返回值

类型	描述
table	表的索引为 1

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFileDialog\脚本\XMain.lua

LXFontDialog

功能描述

实现字体对话

函数列表

函数名称	说明
new	创建一个 LXFontDialog 对象
GetColor	获得文字颜色
GetCurrentFont	获得当前字体
GetFaceName	获得选择字体名称
GetSize	获得字体大小
GetWeight	获得字体磅重
IsBold	是否粗体
IsItalic	是否斜体
IsUnderline	是否下划线

new

创建一个 LXFontDialog 对象

```
new(font, parent)
```

参数

参数名	类型	描述	是否可选	默认值
font	字体 LXFont 对象	选择的默认字体	是	nil
parent	窗口 LXWnd 对象	父窗口	是	nil

返回值

类型	描述
LXFontDialog	创建成功返回一个有效的 LXFontDialog 窗口对象 ,否则返回 nil

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

GetColor

获得文字颜色

GetColor()

参数

无

返回值

类型	描述
LXColor	返回一个 LXColor 颜色对象

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

GetCurrentFont

获得当前字体

GetCurrentFont()

参数

无

返回值

类型	描述
LXFont	返回 LXFont 字体对象

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

GetFaceName

获得选择字体名称

```
GetFaceName()
```

参数

无

返回值

类型	描述
string	返回字体的名称

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

GetSize

获得选择字体大小

```
GetSize()
```

参数

无

返回值

类型	描述
number	字体的大小

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

GetWeight

获得选择字体磅重

GetWeight()

参数

无

返回值

类型	描述
number	字体的磅重

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

IsBold

是否粗体

```
IsBold()
```

参数

无

返回值

类型	描述
boolean	是否粗体

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

IsItalic

是否斜体

```
IsItalic()
```

参数

无

返回值

类型	描述
boolean	是否斜体

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

IsUnderline

是否下划线

IsUnderline()

参数

无

返回值

类型	描述
boolean	是否下划线

注意事项

示例

参见

Samples\Test_Wnd\Test_CommonDlg\Test_LXFontDialog\脚本\XMain.lua

消息对话框

功能描述

在屏幕上显示一个消息对话框

函数列表

函数名称	说明
XMessageBox	在屏幕上显示一个消息对话框

XMessageBox

在屏幕上显示一个消息对话框

```
XMessageBox(strText, strCaption, strBtnType, strIconType, OwnerWnd, hBKImage, hIconImage, hFont)

XMessageBox(strText, strCaption, strBtnType, strIconType, OwnerWnd, strBKFileImage, strIconFileImage, hFont)

XMessageBox(hTextImage, nleftInterval, ntopInterval, strBtnType, strIconType, OwnerWnd, hBKImage, hIconImage, hBtnImage, nBtnInterval, nBtnBottomInterval,)
```

参数

参数名	类型	描述	是否可选	默认值
strText	string	显示内容	否	无
strCaption	string	窗口标题	是	“Msg”
strBtnType	string	按钮类型	是	“OK”
strIconType	string	图标类型	是	“ICONINFORMATION”
OwnerWnd	LXWnd	隶属窗口	是	nil
hBKImage	LXHandle	背景图资源对象	是	nl
hIconImage	LXHandle	自定义图标资源对象	是	nil
strBKFileImage	string	背景图资源文件	是	nil
strIconFileImage	string	图标资源文件	是	nil
hFont	LXFont	绘制文本的字体	是	nil
hTextImage	LXHandle	文本图资源对象	否	无
nleftInterval	number	文本图离图标的横向间隔	是	8
ntopInterval	number	文本图离窗口上面的纵向间隔	是	20
hBtnImage	LXHandle	按钮图资源对象	是	nil
nBtnInterval	number	有二个按钮以上时按钮	是	8

		间的距离		
nBtnBottomInterval	number	按钮离窗口下方的纵向间隔	是	20

返回值

类型	描述
number	对应不同按钮点击返回结果不同 点击了确定将返回 1 点击了取消将返回 2 点击了终止将返回 3 点击了重试将返回 4 点击了忽略将返回 5 点击了是将返回 6 点击了否将返回 7

注意事项

XMessageBox 可创建一个文本形的或文本为图像形的消息对话框

窗口标题 strCaption 默认值”Msg”
按钮类型 strBtnType 取值如下:
“OK” --OK 按钮 默认值
“OKCANCEL” --OK & CANCEL 按钮
“YESNOCANCEL” --YES & NO & CANCEL 按钮
“YESNO” --YES & NO 按钮
“RETRYCANCEL” --RETRY & CANCEL 按钮

图标类型 strIconType 取值如下:
“ICONINFORMATION” --信息提示图标 默认值
“ICONEXCLAMATION” --“!”提示图标
“ICONQUESTION” --“?”提示图标
“ICONSTOP” --停止提示图标
“USERICON” --用户自定义图标 此类型需指定图片资源对象 hIconImage

hBtnImage 为按钮图.是确定,取消,终止,重试,忽略,是和否的 7 个按钮列表图,每个按钮图为四态图

示例

```
XMessageBox("test", "Caption", "OK", "ICONEXCLAMATION")
```

参见

```
Samples\Test_Wnd\Test_CommonDlg\Test_LXMsgBox\脚本\XMain.lua
```

3.6 场景 API

本部分内容介绍 X 魔方编辑器提供的 3 维场景 Lua 导出类，用户使用这些导出类提供的功能可以创建属于自己的虚拟 3 维世界。导出类名称与功能如下表所示:

对象名	功能
LXAABBBox	包围盒，实现对包围盒的创建等操作
LXAABBVolume	包围圆柱，实现对包围圆柱的创建等操作
LXBoundBox	非轴对称包围盒，实现非轴对称包围盒的创建等操作
LXMatrix	矩阵，实现矩阵数学运算
LXPlane	平面，实现平面对象的创建等操作
LXQuaternion	四元数，实现对四元数的数学运算
LXVector2	2 维向量，实现 2 维向量的创建等操作
LXVector3	3 维向量，实现 3 维向量的创建等操作
LXVector4	4 维向量，实现 4 维向量的创建等操作
LXSceneCamera	摄像机对象，实现对相机的操作等功能
LXSceneManager	场景管理器对象，实现管理场景中的相机、物体等功能
LXSceneItem	物体对象，实现创建场景物体，设置物体属性、动画等功能
LXSceneNode	物体节点对象，用于添加\删除节点，设置物体节点属性等功能

3.6.1 基本对象

LXAABBBox

功能描述

该对象实现包围盒功能。

函数列表

函数名称	说明
new	创建一个 LXAABBBox 对象
EmptyAABBBox	清空内部包围盒数据

IsEmpty	包围盒是否为空
SetInfinite	设置内部包围盒为无限大
IsInfinite	包围盒是否为无限大
SetAABBBox	设置包围盒
AddVertex	往包围盒中添加一个顶点
NormalizeAABBBox	正规化包围盒
UpdateMatrix	根据给定的变换矩形变换包围盒对象
UpdateAffineMatrix	应用仿射变换矩阵变换包围盒对象
GetAABBCorner	得到包围盒指定角的位置
GetAABBCorners	得到包围盒所有角的位置数组
GetMin	得到包围盒最小顶点
GetMax	得到包围盒最大顶点
SetMin	设置包围盒最小顶点
SetMax	设置包围盒最大顶点
GetCenter	得到包围盒中心点
GetSize	得到包围盒大小
GetHalfSize	得到包围盒半大小
TestVertexInBox	指定的象素点是否在包围盒内
GetNearsPtOnAABBBox	给定空间点，计算包围盒上离该点最近的点
IntersectAABBBox	得到两个包围盒的交集
UnionAABBBOx	得到两个包围盒的并集
IsIntersect	判断两个包围盒是否相交
IncludeBox	判断当前包围盒是否完全包含指定的包围盒
Scale	缩放当前包围盒
TestAABBBoxIntersectDir	测试包围盒是否和指定的射线相交
TestAABBBoxIntersectDetail	测试包围盒是否和指定的射线相交
TestAABBBoxBoxIntersectPlane	测试包围盒是否和指定的平面相交
TestAABBBoxIntersectTriangle	测试包围盒是否和指定的三角形相交
TestAABBBoxIntersectVolume	测试包围盒是否和包围圆柱相交
TestIntersectMovingVolume	计算与一个运动圆柱的交点
OffsetAABbBox	偏移包围盒

new

创建一个 LXAABBBox 对象

```
IsEmpty()
```

参数

```
无
```

返回值

类型	描述
LXAABBBox	返回一个 LXAABBBox 对象

注意事项

无

示例

```
local    AABBBox = LXAABBBox.new()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

EmptyAABBBox

清空内部包围盒数据

```
EmptyAABBBox()
```

参数

无

返回值

无

注意事项

无

示例

```
AABBBox = LXAABBBox.new();
AABBBox.EmptyAABBBox()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

IsEmpty

包围盒是否为空

```
IsEmpty()
```

参数

无

返回值

类型	描述
boolean	为空返回 true，否则返回 false

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
bEmpty = AABBBox.IsEmpty()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

SetInfinite

设置内部包围盒为无限大

```
SetInfinite()
```

参数

无

返回值

无

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
AABBBox.SetInfinite()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

IsInfinite

包围盒是否为无限大

```
IsInfinite()
```

参数

无

返回值

类型	描述
boolean	是无限大返回 true，否则返回 false

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
bInfinite = AABBBox.IsInfinite()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

SetAABBBox

设置包围盒

```
SetAABBBox(minVec3, maxVec3)
SetAABBBox(fMinX, fMinY, fMinZ, fMaxX, fMaxY, fMaxZ)
```

参数

数名	类型	描述	是否可选	默认值
minVec3	LXVector3	包围盒最小顶点值	否	无
maxVec3	LXVector3	包围盒最大顶点值	否	无
fMinX	number	包围盒最小 x 值	否	无
fMinY	number	包围盒最小 y 值	否	无
fMinZ	number	包围盒最小 z 值	否	无
fMaxX	number	包围盒最大 x 值	否	无
fMaxY	number	包围盒最大 y 值	否	无
fMaxZ	number	包围盒最大 z 值	否	无

返回值

无

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
minVec3 = LXVector3.new();
maxVec3 = LXVector3.new();
minVec3.SetValue(1, 1, 1);
maxVec3.SetValue(5, 6, 7);
AABBBox.SetAABBBox(minVec3, maxVec3);
AABBBox.SetAABBBox(1, 1, 1, 5, 6, 7);
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

AddVertex

往包围盒中添加一个顶点

```
AddVertex(Vec3)
AddVertex(x, y, z)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	顶点	否	无
x	number	顶点 X 坐标	否	无
y	number	顶点 Y 坐标	否	无
z	number	顶点 Z 坐标	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new();
```

```
Vec3.SetValue(1, 1, 1);
AABBBox = LXAABBBox.new()
AABBBox.AddVertex(Vec3)
AABBBox.AddVertex(1, 1, 1)
```

参见

```
Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua
```

AddAABBBox

往包围盒中添加一个包围盒

```
AddAABbBox(aabbbox)
```

参数

参数名	类型	描述	是否可选	默认值
aabbbox	LXAABBBox	包围盒	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Aabbbox1 = LXAABBBox.new();
Aabbbox1.SetAABBBox(1, 1, 1, 3, 5, 6)
AABBBox = LXAABBBox.new()
AABBBox.AddAABbBox(Aabbbox1);
```

参见

```
Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua
```

NormalizeAABBBox

正规化包围盒

```
NormalizeAABBBox()
```

参数

```
无
```

返回值

无

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
AABBBox.NormalizeAABBBox()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

UpdateMatrix

根据给定的变换矩形变换包围盒对象

参数

参数名	类型	描述	是否可选	默认值
Matrix	LXMatrix	变换矩阵	否	无

返回值

无

注意事项

无

示例

```
Matrix = LXMatrix.new()
Matrix.SetRow(1, 1, 2, 3, 4)
AABBBox = LXAABBBox.new()
AABBBox.UpdateMatrix(Matrix)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

UpdateAffineMatrix

应用仿射变换矩阵变换包围盒对象

```
UpdateAffineMatrix(Matrix)
```

参数

参数名	类型	描述	是否可选	默认值
Matrix	LXMatrix	变换矩阵	否	无

返回值

无

注意事项

无

示例

```
Matrix = LXMatrix.new()
Matrix.SetRow(1, 1, 2, 3, 4)
AABBBox = LXAABBBox.new()
AABBBox.UpdateAffineMatrix(Matrix)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

GetAABBCorner

得到包围盒指定角的位置

```
GetAABBCorner(nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	索引	否	无

返回值

类型	描述
LXVector3	LXVector3 导出类类型

注意事项

nIndex 取值范围为 0~7

示例

```
AABBBox = LXAABBBox.new()
```

```
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
Vec3 = AABBBox.GetAABBCorner(0)
```

参见

```
Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua
```

GetAABBCorners

得到包围盒所有角的位置数组

```
GetAABBCorners()
```

参数

```
无
```

返回值

类型	描述
Table	LXVector3 导出类类型数组

注意事项

```
无
```

示例

```
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
ArrVec3 = AABBBox.GetAABBCorners()
```

参见

```
Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua
```

GetMin

得到包围盒最小顶点

```
GetMin()
```

参数

```
无
```

返回值

类型	描述
LXVector3	LXVector3 导出类类型

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
Vec3 = AABBBox.GetMin()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

GetMax

得到包围盒最大顶点

```
GetMax()
```

参数

无

返回值

类型	描述
LXVector3	LXVector3 导出类类型

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
Vec3 = AABBBox.GetMax()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

SetMin

设置包围盒最小顶点

```
SetMin(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	顶点	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new()
Vec3.SetValue(1, 1, 1)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
AABBBox.SetMin(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

SetMax

设置包围盒最大顶点

```
SetMax(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	顶点	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new()
Vec3.SetValue(5, 6, 7)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
AABBBox.SetMax(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

GetCenter

得到包围盒中心点

```
GetCenter()
```

参数

无

返回值

类型	描述
LXVector3	LXVector3 导出类类型

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
Vec3 = AABBBox.GetCenter()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

GetSize

得到包围盒大小

```
GetSize()
```

参数

无

返回值

类型	描述
LXVector3	LXVector3 导出类类型

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
Vec3 = AABBBox.GetSize()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

GetHalfSize

得到包围盒半大小

```
GetHalfSize()
```

参数

无

返回值

类型	描述
LXVector3	LXVector3 导出类类型

注意事项

无

示例

```
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
Vec3 = AABBBox.GetHalfSize()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

TestVertexInBox

指定的象素点是否在包围盒内

TestVertexInBox(Vec3)

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	顶点	否	无

返回值

类型	描述
boolean	在包围盒返回 true，否则返回 false

注意事项

无

示例

```
Vec3 = LXVector3.new()
Vec3.SetValue(1, 2, 3)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bInBox = AABBBox.TestVertexInBox(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

GetNearstPtOnAABBBox

给定空间点，计算包围盒上离该点最近的点

GetNearstPtOnAABBBox(Vec3)

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	顶点	否	无

返回值

类型	描述
LXVector3	包围盒上离该点最近的点

注意事项

无

示例

```
Vec3 = LXVector3.new()
Vec3.SetValue(1, 2, 3)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
retVec3 = AABBBox.GetNearsPtOnAABBBox(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

InterSectAABBBox

得到两个包围盒的交集

```
InterSectAABBBox(Aabbbox1, Aabbbox2)
```

参数

参数名	类型	描述	是否可选	默认值
Aabbbox1	LXAABBBox	包围盒 1	否	无
Aabbbox2	LXAABBBox	包围盒 2	否	无

返回值

类型	描述
Boolean	相交返回 true，否则返回 false

注意事项

无

示例

```
Aabbbox1 = LXAABBBox.new();
Aabbbox2 = LXAABBBox.new();
Aabbbox1.SetAABBBox(1, 1, 1, 5, 6, 7)
Aabbbox2.SetAABBBox(2, 2, 2, 3, 4, 5)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bIntersect = AABBBox.InterSectAABBBox(Aabbbox1, Aabbbox2)
```

参见

```
Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua
```

UnionAABBBox

得到两个包围盒的并集

```
UnionAABBBox(Aabbbox1, Aabbbox2)
```

参数

参数名	类型	描述	是否可选	默认值
Aabbbox1	LXAABBBox	包围盒 1	否	无
Aabbbox2	LXAABBBox	包围盒 2	否	无

返回值

类型	描述
Boolean	并集成功返回 true，否则返回 false

注意事项

无

示例

```
Aabbbox1 = LXAABBBox.new();
Aabbbox2 = LXAABBBox.new();
Aabbbox1.SetAABBBox(1, 1, 1, 5, 6, 7)
Aabbbox2.SetAABBBox(2, 2, 2, 3, 4, 5)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bUnionSuc = AABBBox.UnionAABBBox(Aabbbox1, Aabbbox2)
```

参见

```
Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua
```

IsInterSect

判断两个包围盒是否相交

```
IsInterSect(Aabbbox)
```

参数

参数名	类型	描述	是否可选	默认值
Aabbbox	LXAABBBox	包围盒	否	无

返回值

类型	描述
Boolean	相交返回 true，否则返回 false

注意事项

无

示例

```
Aabbbox1 = LXAABBBox.new();
Aabbbox1.SetAABBBox(1, 1, 1, 5, 6, 7)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bInterSect = AABBBox.IsInterSect(Aabbbox1)
```

参见

```
Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua
```

IncludeBox

判断当前包围盒是否完全包含指定的包围盒

```
IncludeBox(Aabbbox)
```

参数

参数名	类型	描述	是否可选	默认值
Aabbbox	LXAABBBox	包围盒	否	无

返回值

类型	描述
Boolean	完全包含返回 true，否则返回 false

注意事项

无

示例

```
Aabbbox1 = LXAABBBox.new();
Aabbbox1.SetAABBBox(1, 1, 1, 5, 6, 7)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bInclude = AABBBox.IncludeBox(Aabbbox1)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

Scale

缩放当前包围盒

```
Scale(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	缩放因子	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new()
Vec3.SetValue(1, 1, 1)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
AABBBox.Scale(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

TestAABBBoxIntersectDir

测试包围盒是否和指定的射线相交,返回原点和交点距离如果不相交返回<0.0f

```
TestAABBBoxIntersectDir(orgVec3, dirVec3, fMinLimit,    bDiscardInsides)
```

参数

参数名	类型	描述	是否可选	默认值
orgVec3	LXVector3	原点位置	否	无
dirVec3	LXVector3	方向位置	否	无
fMinLimit	number	最小限制距离	否	无
bDiscardInsides	boolean	在包围盒内部是否认为不相交	否	无

返回值

类型	描述
boolean	相交返回 true，否则返回 false
LXVector3	相交返回相交点，否则返回 nil 值

注意事项

无

示例

```
orgVec3 = LXVector3.new()
orgVec3.SetValue(1, 1, 1)
dirVec3 = LXVector3.new();
dirVec3.SetValue(2, 3, 4)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bIntersect, retVec3 = AABBBox.TestAABBBoxIntersectDir(
orgVec3, dirVec3, 5.0, false)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

TestAABBBoxIntersectDirDetail

测试包围盒是否和指定的射线相交

```
TestAABBBoxIntersectDirDetail(orgVec3, dirVec3, fMinLimit, fMinDistance, fMaxDistance, bDiscardInsides)
```

参数

参数名	类型	描述	是否可选	默认值
orgVec3	LXVector3	原点位置	否	无
dirVec3	LXVector3	方向值	否	无
fMinLimit	number	最小限制距离	否	无
fMinDistance	number	最小距离	否	无
fMaxDistance	number	最大距离	否	无
bDiscardInsides	boolean	在包围盒内部是否认为不相交	否	无

返回值

类型	描述
boolean	相交返回 true，否则返回 false
number	相交返回最小的距离，否则返回 nil
number	相交返回最大距离，否则返回 nil

注意事项

无

示例

```
orgVec3 = LXVector3.new()
orgVec3.SetValue(1, 1, 1)
dirVec3 = LXVector3.new();
dirVec3.SetValue(2, 3, 4)
intersectVec3 = LXVector3.new();
intersectVec3.SetValue(5, 6, 7)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bIntersect, fMinDist, fMaxDist =  AABBBox. TestAABBBoxIntersectDirDetail ( orgVec3, dirVec3, 5.0, 2.0, 9.0, false)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

TestAABBBoxBoxIntersectPlane

测试包围盒是否和指定的平面相交

```
TestAABBBoxBoxIntersectPlane(plane)
```

参数

参数名	类型	描述	是否可选	默认值
Plane	LXPlane	平面	否	无

返回值

类型	描述
boolean	相交返回 true，否则返回 false

注意事项

无

示例

```
Plane = LXPlane.new()
Vec3 = LXVector3.new()
Vec3.SetValue(1, 1, 1)
NormalVec3 = LXVector3.new();
NormalVec3.SetValue(2, 3, 4)
Plane. CreatePlaneFromPointNormal(Vec3, NormalVec3)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bIntersect = AABBBox. TestAABBBoxBoxIntersectPlane(Plane)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

TestAABBBoxIntersectTriangle

测试包围盒是否和指定的三角形相交

```
TestAABBBoxIntersectTriangle(vec31, vec32, vec33)
```

参数

参数名	类型	描述	是否可选	默认值
Vec31	LXVector3	三角形顶点 1	否	无
Vec32	LXVector3	三角形顶点 2	否	无
Vec33	LXVector3	三角形顶点 3	否	无

返回值

类型	描述
boolean	相交返回 true，否则返回 false

注意事项

无

示例

```
Vec31 = LXVector3.new()
Vec31.SetValue(1, 1, 1)
Vec32 = LXVector3.new()
Vec32.SetValue(2, 2, 2)
Vec33 = LXVector3.new()
Vec33.SetValue(3, 3, 3)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bIntersect = AABBBox.TestAABBBoxIntersectTriangle(Vec31, Vec32, Vec33)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

TestAABBBoxIntersectVolume

测试包围盒是否和包围圆柱相交

```
TestAABBBoxIntersectVolume(Volume)
```

参数

参数名	类型	描述	是否可选	默认值
Volume	LXAABBVolume	包围圆柱	否	无

返回值

类型	描述

boolean	相交返回 true，否则返回 false

注意事项

无

示例

```
Volume = LXAABBVolume.new()
Vec3 = LXVector3.new()
Vec3.SetValue(1, 1, 1)
Volume.InitAABBVolume(vec3, 2.0, 5.0)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bIntersect = AABBBox.TestAABBBoxIntersectVolume(Volume)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

TestIntersectMovingVolume

计算与一个运动圆柱的交点,存在交点返回距离<0.0f 无交点

```
TestIntersectMovingVolume(Volume,dirVec3,fLimit,bCollideInBox)
```

参数

参数名	类型	描述	是否可选	默认值
Volume	LXAABBVolume	运动圆柱	否	无
dirVec3	LXVector3	方向值	否	无
fLimit	number	限定的距离	否	无
bCollideInBox	boolean	在包围盒里面是否认为碰撞	否	无

返回值

类型	描述
boolean	相交返回 true，否则返回 false
LXVector3	相交返回交点，否则返回 nil

注意事项

无

示例

```
Volume = LXAABBVolume.new()
Vec3 = LXVector3.new()
Vec3.SetValue(1, 1, 1)
Volume.InitAABBVolume(vec3, 2.0, 5.0)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
bIntersect, intersectVec3 = AABBBox. TestIntersectMovingVolume(Volume, vec3, 5.0, true)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

OffsetAABbBox

偏移包围盒

```
OffsetAABbBox(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	偏移值	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new()
Vec3.SetValue(1, 1, 1)
AABBBox = LXAABBBox.new()
AABBBox.SetAABBBox(1, 1, 1, 3, 5, 6)
AABBBox.OffsetAABbBox(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_LXAABBBox.lua

LXAABBVolume

功能描述

该对象实现包围圆柱功能。

函数列表

函数名称	说明
new	创建一个 LXAABBVolume 对象
InitAABBVolume	初始化包围圆柱对象
TestVertexInVolume	指定的象素点是否在包围圆柱内
GetBottomCenterPt	得到包围圆柱底边中心点坐标坐标
GetRadius	得到包围圆柱圆柱半径
GetHeight	得到包围圆柱圆柱高度
TestIntersectVolume	测试是否和包围圆柱相交
TestIntersectMoveingVolume	计算与一个运动圆柱的交点
UpdateMatrix	根据给定的变换矩阵变换包围圆柱
SetScaleVolumn	设置给定的缩放变换包围圆柱

new

创建一个 LXAABBVolume 对象

```
IsEmpty()
```

参数

无

返回值

类型	描述
LXAABBVolume	返回一个 LXAABBVolume 对象

注意事项

无

示例

```
local    AABBVol = LXAABBVolume.new()
```

参见

Samples\Test_LuaBase\Test_LXAABBBox\Test_AABBVolume.lua

InitAABBVolume

根据给定的底边中心点坐标坐标,圆柱半径,圆柱高度，初始化包围圆柱对象

InitAABBVolume(Vec3, radius, height)

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	底边中心点坐标	否	无
Radius	number	圆柱半径	否	无
Height	number	圆柱高度	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new(1, 2, 3)
AABBVolume = LXAABBVolume.new()
AABBVolume.InitAABBVolume(Vec3, 2.8, 3.0)
```

参见

Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua

TestVertexInVolume

指定的象素点是否在包围圆柱内

TestVertexInVolume (Vec3)

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	像素点	否	无

返回值

类型	描述

boolean	在圆柱内返回 true，否则返回 false

注意事项

无

示例

```
Vec3 = LXVector3.new(1, 2, 3)
AABBVolume = LXAABBVolume.new()
AABBVolume.TestVertexInVolume (Vec3)
```

参见

Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua

GetBottomCenterPt

得到包围圆柱底边中心点坐标坐标

```
GetBottomCenterPt()
```

参数

无

返回值

类型	描述
LXVector3	LXVector3 导出类类型

注意事项

无

示例

```
AABBVolume = LXAABBVolume.new()
Vec3 = AABBVolume.GetBottomCenterPt()
```

参见

Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua

GetRadius

得到包围圆柱圆柱半径

```
GetRadius()
```

参数

```
无
```

返回值

类型	描述
number	圆柱半径

注意事项

```
无
```

示例

```
AABBVolume = LXAABBVolume.new()
Radius = AABBVolume.GetRadius()
```

参见

```
Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua
```

GetHeight

得到包围圆柱圆柱高度

```
GetHeight()
```

参数

```
无
```

返回值

类型	描述
number	圆柱高度

注意事项

```
无
```

示例

```
AABBVolume = LXAABBVolume.new()
Height = AABBVolume.GetHeight()
```

参见

Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua

TestIntersectVolume

测试是否和包围圆柱相交

TestIntersectVolume(Aabbvolume)

参数

参数名	类型	描述	是否可选	默认值
Aabbvolume	LXAABBVolume	包围圆柱	否	无

返回值

类型	描述
boolean	相交返回 true，否则返回 false

注意事项

无

示例

```
Aabbvol = LXAABBVolume.new()
Vec3 = LXVector3.new(1, 2, 3)
Aabbvol.InitAABBVolume(vec3, 2.0, 3.0);
AABBVolume = LXAABBVolume.new()
AABBVolume.TestIntersectVolume(Aabbvol)
```

参见

Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua

TestIntersectMovingVolume

计算与一个运动圆柱的交点,存在交点返回距离和交点, <0.0f 无交点

TestIntersectMovingVolume(Aabbvol, Vec3, fLimit)

参数

参数名	类型	描述	是否可选	默认值
Aabbvol	LXAABBVolume	运动圆柱	否	无
Vec3	LXVector3	方向	否	无

fLimit	number	限定范围	否	无

返回值

类型	描述
number	与交点的距离
LXVector3	相交返回交点位置，否则返回 nil

注意事项

无

示例

```
Aabbvol = LXAABBVolume.new()
Vec3 = LXVector3.new(1, 2, 3)
Aabbvol.InitAABBVolume(vec3, 2.0, 3.0);
AABBVolume = LXAABBVolume.new()
fLimit, vec3 = AABBVolume. TestIntersectMovingVolume(Aabbvol, Vec3, 5.2)
```

参见

Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua

UpdateMatrix

根据给定的变换矩阵变换包围圆柱

```
UpdateMatrix(Matrix)
```

参数

参数名	类型	描述	是否可选	默认值
Matrix	LXMatrix	变换矩阵	否	无

返回值

类型	描述
Boolean	成功变换返回 true，失败返回 false

注意事项

无

示例

```
Matrix = LXMatrix.new();
Matrix.SetRow(1, 1, 2, 3, 4)
AABBVolume = LXAABBVolume.new()
bSuc = AABBVolume.UpdateMatrix(Matrix)
```

参见

```
Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua
```

SetScaleVolumn

设置给定的缩放变换包围圆柱

```
SetScaleVolumn(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	缩放因子	否	无

返回值

类型	描述
boolean	成功返回 true，失败返回 false

注意事项

```
无
```

示例

```
Vec3 = LXVector3.new()
Vec3. SetValue(1, 2, 3);
AABBVolume = LXAABBVolume.new()
bSuc = AABBVolume.SetScaleVolumn(Vec3)
```

参见

```
Samples\Test_LuaBase\Test_AABBVolume\Test_AABBVolume.lua
```

LXBoundBox

功能描述

该对象实现非轴对齐的包围盒功能。

函数列表

函数名称	说明
new	创建一个 LXBoundBox
InitX3DBoundBox	初始化包围盒
UpdateMatrix	根据给定的变换矩形变换包围盒对象
TestBoundBoxIntersectVolume	测试包围盒是否和包围圆柱相交
TestBoundBoxIntersectDir	测试包围盒是否和指定的射线相交
TestBoundBoxIntersectTriangle	测试包围盒是否和指定的三角形相交
GetBoundBoxCenter	计算包围盒中心点
TestVertexInBoundBox	指定的象素点是否在包围盒内
GetBoundBoxPlane	得到包围盒 6 个相关平面
GetBoundBoxTriangleIndex	得到包围盒 6 个平面的三角形索引
GetBoundBoxLineIndex	得到包围盒 6 个平面的线索引
GetAABBBox	得到内部对应的轴对称包围盒
GetContainerAABBBox	得到包容该非轴对齐的包围盒的最小包围盒
GetVertex	得到指定索引位置的顶点

new

创建一个 LXBoundBox 对象

```
IsEmpty()
```

参数

无

返回值

类型	描述
LXBoundBox	返回一个 LXBoundBox 对象

注意事项

无

示例

```
local    BoundBox = LXBoundBox.new()
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

InitX3DBoundBox

```
InitX3DBoundBox()      初始化非轴对齐的包围盒
InitX3DBoundBox(Aabbbox, Matrix)  根据轴对称包围盒初始化
```

参数

参数名	类型	描述	是否可选	默认值
Aabbbox	LXAABBBox	包围盒	否	无
Matrix	LXMatrix	变换矩阵	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
BoundBox = LXBoundBox.new()
BoundBox.InitX3DBoundBox()
Aabbbox = LXAABBBox.new()
Aabbbox.SetAABBBox(1,2, 3, 4, 5, 6)
Matrix = LXMatrix.new()
Matrix.SetRow(1, 1, 2, 3, 4)
BoundBox.InitX3DBoundBox(Aabbbox, Matrix)
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

UpdateMatrix

根据给定的变换矩形变换包围盒对象

```
UpdateMatrix(Matrix)
```

参数

参数名	类型	描述	是否可选	默认值
Matrix	LXMatrix	变换矩阵	否	无

返回值

类型	描述
boolean	成功变换返回 true，否则返回 false

注意事项

无

示例

```
Matrix = LXMatrix.new();
Matrix.SetRow(1, 1, 2, 3, 4)
BoundBox = LXBoundBox.new()
bSuc = BoundBox.UpdateMatrix(Matrix)
```

参见

Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua

TestBoundBoxIntersectVolume

测试包围盒是否和包围圆柱相交

```
TestBoundBoxIntersectVolume(Volume)
```

参数

参数名	类型	描述	是否可选	默认值
Volume	LXAABBVolume	包围圆柱	否	无

返回值

类型	描述
Boolean	相交返回 true，否则返回 false

注意事项

无

示例

```
Volume = LXAABBVolume.new()
Vec3 = LXVector3.new()
Vec3.SetValue(1, 2, 3)
Volue.InitAABBVolue(vec3, 2, 4)
BoundBox = LXBoundBox.new()
bIntersect = BoundBox. TestBoundBoxIntersectVolume(Volume)
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

TestBoundBoxIntersectDir

测试包围盒是否和指定的射线相交,返回原点和交点距离如果不相交返回<0.0f

```
TestBoundBoxIntersectDir(orgVec3, dirVec3, fMinLimit)
```

参数

参数名	类型	描述	是否可选	默认值
orgVec3	LXVector3	射线起点	否	无
dirVec3	LXVector3	射线方向	否	无
fMinLimit	number	限制的距离	否	无

返回值

类型	描述
Boolean	相交返回 true，否则返回 false
LXVector3	相交返回交点，否则返回 nil 值

注意事项

```
无
```

示例

```
orgVec3 = LXVector3.new()
orgVec3.SetValue(1, 1, 1)
dirVec3 = LXVector3.new();
dirVec3.SetValue(2, 3, 4)
intersectVec3 = LXVector3.new();
intersectVec3.SetValue(5, 6, 7)
BoundBox = LXBoundBox.new()
```

```
bIntersect, retVec3 = BoundBox. TestBoundBoxIntersectDir(orgVec3, dirVec3, 5.0)
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

TestBoundBoxIntersectTriangle

测试包围盒是否和指定的三角形相交

```
TestBoundBoxIntersectTriangle(Vec31, Vec32, Vec33)
```

参数

参数名	类型	描述	是否可选	默认值
Vec31	LXVector3	三角形顶点 1	否	无
Vec32	LXVector3	三角形顶点 2	否	无
Vec33	LXVector3	三角形顶点 3	否	无

返回值

类型	描述
boolean	相交返回 true，失败返回 false

注意事项

```
无
```

示例

```
Vec31 = LXVector3.new()
Vec31.SetValue(1, 1, 1)
Vec32 = LXVector3.new()
Vec32.SetValue(2, 2, 2)
Vec33 = LXVector3.new()
Vec33.SetValue(3, 3, 3)
BoundBox = LXBoundBox.new()
bIntersect = BoundBox.TestBoundBoxIntersectTriangle (Vec31, Vec32, Vec33)
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

GetBoundBoxCenter

计算包围盒中心点

```
GetBoundBoxCenter()
```

参数

无

返回值

类型	描述
LXVector3	包围盒中心点

注意事项

无

示例

```
Vec3 = BoundBox.GetBoundBoxCenter()
```

参见

Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua

TestVertexInBoundBox

指定的象素点是否在包围盒内

```
TestVertexInBoundBox(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	像素点	否	无

返回值

类型	描述
Boolean	成功返回 true，失败返回 false

注意事项

无

示例

```
Vec3 = LXVector3.new()
Vec3. SetValue(1, 2, 3)
BoundBox = LXBoundBox.new()
bInBox = BoundBox.TestVertexInBoundBox(Vec3)
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

GetBoundBoxPlane

得到包围盒 6 个相关平面,pPlane 必须分配项,返回后函数顺序 near far left right top bottom

```
GetBoundBoxPlane()
```

参数

```
无
```

返回值

类型	描述
Table	包围盒相关的 6 个平面

注意事项

```
无
```

示例

```
BoundBox = LXBoundBox.new()
PlaneTable = BoundBox.GetBoundBoxPlane()
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

GetBoundBoxTriangleIndex

得到包围盒 6 个平面的三角形索引,总共个角形 36 个索引项,事先必须分配内存

```
GetBoundBoxTriangleIndex(index)
```

参数

参数名	类型	描述	是否可选	默认值
index	number 类型数组		否	无

返回值

无

注意事项

无

示例

```
for i = 1, 36 do
index[i] = i
End
BoundBox = LXBoundBox.new()
BoundBox.GetBoundBoxTriangleIndex(index)
```

参见

Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua

GetBoundBoxLineIndex

得到包围盒 6 个平面的边索引,总共条边 24 个索引项,事先必须分配内存

```
GetBoundBoxLineIndex(index)
```

参数

参数名	类型	描述	是否可选	默认值
index	number 类型数组		否	无

返回值

无

注意事项

无

示例

```
for i = 1, 24 do
```

```
index[i] = i
End
BoundBox = LXBoundBox.new()
BoundBox.GetBoundBoxLineIndex (index)
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

GetAABBBox

得到内部对应的轴对称包围盒

```
GetAABBBox()
```

参数

```
无
```

返回值

类型	描述
LXAABBBox	内部对应的轴对称包围盒

注意事项

```
无
```

示例

```
BoundBox = LXBoundBox.new()
Aabbbox = BoundBox.GetAABBBox()
```

参见

```
Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua
```

GetContainerAABBBox

得到包容该非轴对齐的包围盒的最小包围盒

```
GetContainerAABBBox()
```

参数

```
无
```

返回值

类型	描述
LXAABBBox	包容该非轴对齐的包围盒的最小包围盒

注意事项

无

示例

```
BoundBox = LXBoundBox.new()
Aabbbox = BoundBox.GetContainerAABBBox()
```

参见

Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua

GetVertex

得到指定索引位置的顶点

```
GetVertex(nIndex)
```

参数

参数名	类型	描述	是否可选	默认值
nIndex	number	索引值	否	无

返回值

类型	描述
LXVector3	指定索引位置的顶点

注意事项

无

示例

```
BoundBox = LXBoundBox.new()
Vec3 = BoundBox.GetVertex(1)
```

参见

Samples\Test_LuaBase\Test_LXBoudBox\Test_LXBoundBox.lua

LXMatrix

功能描述

本类是对矩阵的数学运算及操作的封装类。

函数列表

函数名称	说明
new	新建一个 LXMatrix 对象
SetRow	设置矩阵某一行的值
GetRow	得到矩阵某一列的值
SetCol	设置矩阵某一列的值
GetCol	得到矩阵某一列的值
SetIdentity	设置单位矩阵
IsIdentity	指定矩阵是否是单位矩阵
Transpose	计算转置矩阵
FromCubeMapViewMatrix	得到渲染立方体纹理指定面的视角矩阵
MakeTransform	根据给定的缩放平移旋转构造矩阵
MakeTransformInverse	根据给定的缩放平移旋转构造矩阵的逆矩阵
AffineTransformation	建造一个 3D 放射转换矩阵
AffineTransformation2D	在 xy 平面上建造一个 D 放射转换矩阵
Decompose	求出当前矩阵的秩
Determinant	求出当前矩阵的秩
Inverse	求出当前矩阵的逆矩阵
LookAtLH	建立左手坐标系的 Look-At 矩阵
Multiply	矩阵相乘
MultiplyTranspose	计算矩阵相乘的结果的转置矩阵
OrthoLH	建立左手坐标系下的正交透视矩阵
OrthoOffCenterLH	建立左手坐标系下的定制正交透视矩阵
PerspectiveFovLH	建立左手坐标下透视矩阵
PerspectiveLH	建立左手坐标系下的透视矩阵
PerspectiveOffCenterLH	建立左手坐标系下的定制透视矩阵
RotationAxis	建立一个绕任意轴旋转的矩阵
RotationQuaternion	根据一个四元数创建旋转矩阵
Rotation	建立绕指定轴旋转矩阵
Scaling	建立一个缩放矩阵
Translation	建立一个位移矩阵
Transform Vec2Coord	使用当前矩阵转换指定维向量
Transform Vec2Normal	使用当前矩阵转换指定维向量

Transform Vec3Coord	使用当前矩阵转换指定维向量
Transform Vec3Normal	使用当前矩阵转换指定维向量

new

创建一个 LXMatrix 对象

```
new()
```

参数

无

返回值

类型	描述
LXMatrix	LXMatrix 对象

注意事项

无

示例

```
LMtx = LXMatrix.new()
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

SetRow

设置矩阵某一行的值

```
SetRow(Row,f1,f2,f3,f4)
```

参数

参数名	类型	描述	是否可选	默认值
Row	number	指定行	否	无
f1	number	数据 1	否	无
f2	number	数据 2	否	无
f3	number	数据 3	否	无
f4	number	数据 4	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.SetRow (1, 1,2,3,4)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

GetRow

得到矩阵某一行的值

```
GetRow(Row)
```

参数

参数名	类型	描述	是否可选	默认值
Row	number	指定行	否	无

返回值

类型	描述
number	数据 1
number	数据 2
number	数据 3
number	数据 4

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.SetRow (2, 1,2,3,4)
f1,f2,f3,f4 = LMtx.GetRow (2)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

SetCol

设置矩阵某一列的值

SetCol(Col,f1,f2,f3,f4)

参数

参数名	类型	描述	是否可选	默认值
Col	number	指定列	否	无
f1	number	数据 1	否	无
f2	number	数据 2	否	无
f3	number	数据 3	否	无
f4	number	数据 4	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.SetCol(1, 1, 2, 3, 4)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

GetCol

得到矩阵某一列的值

GetCol(Col)

参数

参数名	类型	描述	是否可选	默认值
Col	number	指定列	否	无

返回值

类型	描述
number	数据 1
number	数据 2
number	数据 3
number	数据 4

注意事项

无

示例

```
LMtx = LXMatrix.new()
f1,f2,f3,f4 = LMtx.GetCol (2)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

SetIdentity

设置单位矩阵

```
SetIdentity()
```

参数

无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.SetIdentity()
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

IsIdentity

指定矩阵是否是单位矩阵

```
IsIdentity()
```

参数

无

返回值

类型	描述
Boolean	是否为单位矩阵

注意事项

无

示例

```
LMtx = LXMatrix.new()
bResult = LMtx.IsIdentity ()
if bResult == true then
    print(“是单位阵”)
else
    print(“不是单位阵”)
end
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

Transpose

计算自身转置矩阵

```
Transpose()
```

参数

无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.Transpose()
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

FromCubeMapView

得到渲染立方体纹理指定面的视角矩阵

```
FromCubeMapView(Face)
```

参数

参数名	类型	描述	是否可选	默认值
Face	number	指定纹理面	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
LMtx = LXMatrix.new()
LMtx.FromCubeMapView(3)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

MakeTransform

根据给定的缩放、平移、旋转构造矩阵

```
MakeTransform(vcScale,qRotation,vcTranslation)
```

参数

参数名	类型	描述	是否可选	默认值
vcScale	LXVector3	缩放向量	否	无
qRotation	LXQuaternion	旋转四元素	否	无
vcTranslation	LXVector3	平移向量	否	无

返回值

无

注意事项

操作顺序为：先缩放再旋转再平移

示例

```
LMtx = LXMatrix.new()
vcScale = LXVector3.new(1, 2, 3)
vcTranslation = LXVector3.new(3, 4, 5)
qRotation = LXQuaternion.new(0, 0, 0, 1.0);
LMtx.MakeTransform(vcScale,qRotation,vcTranslation)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

MakeTransformInverse

根据给定的缩放、平移、旋转构造矩阵的逆矩阵

```
MakeTransformInverse(vcScale,qRotation,vcTranslation)
```

参数

参数名	类型	描述	是否可选	默认值
vcScale	LXVector3	缩放向量	否	无
qRotation	LXQuaternion	旋转四元素	否	无
vcTranslation	LXVector3	平移向量	否	无

返回值

无

注意事项

原矩阵构造顺序先缩放再旋转再平移

示例

```
LMtx = LXMatrix.new()
vcScale = LXVector3.new(1, 2, 3)
vcTranslation = LXVector3.new(3, 4, 5)
```

```
qRotation = LXQuaternion.new(0, 0, 0, 1.0);
LMtx.MakeTransformInverse(vcScale,qRotation,vcTranslation)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

AffineTransformation

建造一个 3D 放射转换矩阵

```
AffineTransformation(fScale,RotateCenter, pRotate,pTranslation)
```

参数

参数名	类型	描述	是否可选	默认值
fScale	number	缩放系数	否	无
RotateCenter	LXVector3	旋转中心	否	无
pRotate	LXQuaternion	旋转四元素	否	无
pTranslation	LXVector3	平移向量	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
LMtx = LXMatrix.new()
qRotationCenter = LXVector3.new(1, 2, 3)
pTranslation= LXVector3.new(3, 4, 5)
pRotate = LXQuaternion.new(0, 0, 0, 1.0);
LMtx.AffineTransformation(2.0,pRotationCenter,10.0,pTranslation)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

AffineTransformation2D

在 xy 平面上建造一个 2D 放射转换矩阵

```
AffineTransformation2D (fScale,RotateCenter,pRotate,pTranslation)
```

参数

参数名	类型	描述	是否可选	默认值
fScale	number	缩放系数	否	无
RotateCenter	LXVector2	旋转中心	否	无
pRotate	LXQuaternion	旋转四元素	否	无
pTranslation	LXVector2	平移向量	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
qRotationCenter = LXVector3.new(1, 2)
pTranslation= LXVector3.new(3, 4, 5)
LMtx.AffineTransformation2D (2.0,pRotationCenter,1.0,pTranslation)
```

参见

无

Decompose

把矩阵分解为各转换分量--- 位移 旋转 缩放

```
Decompose()
```

参数

无

返回值

类型	描述
LXVector3	缩放分量
LXQuaternion	旋转分量
LXVector3	平移分量

注意事项

无

示例

```
LMtx = LXMatrix.new()
pOutScale , pOutRotation, pOutTranslation = LMtx.Decompose ()
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

Determinant

求出当前矩阵的秩

```
Determinant()
```

参数

```
无
```

返回值

类型	描述
number	矩阵的秩

注意事项

```
无
```

示例

```
LMtx = LXMatrix.new()
LMtx.Determinant()
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

Inverse

求出当前矩阵的逆矩阵

```
Inverse ()
```

参数

```
无
```

返回值

类型	描述
Boolean	是否存在逆矩阵

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.Inverse ()
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

LookAtLH

建立左手坐标系的 Look-At 矩阵

```
LookAtLH(vcEye,vcAt,vcUp)
```

参数

参数名	类型	描述	是否可选	默认值
vcEye	LXVector3	相机坐标	否	无
vcAt	LXVector3	相机朝向方向向量	否	无
vcUp	LXVector3	上方向向量	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
vcEye = LXVector3.new(1, 2, 3);
vcAt = LXVector3.new(0, 0, 0);
vcUp = LXVector3.new(1, 0, 0);
LMtx.LookAtLH(vcEye,vcAt,vcUp)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

Multiply

矩阵相乘

```
Multiply(Mtx1,Mtx2)
```

参数

参数名	类型	描述	是否可选	默认值
Mtx1	LXMatrix	矩阵 1	否	无
Mtx2	LXMatrix	矩阵 2	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
LMtx = LXMatrix.new()
Mtx1 = LXMatrix.new()
Mtx2 = LXMatrix.new()
Mtx1.SetRow(1, 1, 2, 3, 5)
Mtx2.SetRow(2, 2, 3, 1, 4)
LMtx.Multiply(Mtx1,Mtx2)
```

参见

```
无
```

MultiplyTranspose

计算矩阵相乘的结果的转置矩阵

```
MultiplyTranspose (Mtx1,Mtx2)
```

参数

参数名	类型	描述	是否可选	默认值
Mtx1	LXMatrix	矩阵 1	否	无
Mtx2	LXMatrix	矩阵 2	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
Mtx1 = LXMatrix.new()
Mtx2 = LXMatrix.new()
Mtx1.SetRow(1, 1, 2, 3, 5)
Mtx2.SetRow(2, 2, 3, 1, 4)
LMtx.MultiplyTranspose (Mtx1,Mtx2)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

OrthoLH

建立左手坐标系下的正交透视矩阵

```
OrthoLH(w,h,zn,zf)
```

参数

参数名	类型	描述	是否可选	默认值
w	number	视区卷宽度	否	无
h	number	视区卷高度	否	无
zn	number	视区卷最近面的 Z 值	否	无
zf	number	视区卷最远面的 Z 值	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.OrthoLH(1，2，3，4)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

OrthoOffCenterLH

建立左手坐标系下的定制正交透视矩阵

```
OrthoOffCenterLH(l,r,b,t,zn,zf)
```

参数

参数名	类型	描述	是否可选	默认值
l	number	视区卷最小 X 值	否	无
r	number	视区卷最大 X 值	否	无
b	number	视区卷最小 Y 值	否	无
t	number	视区卷最大 Y 值	否	无
zn	number	视区卷最小 Z 值	否	无
zf	number	视区卷最大 Z 值	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
LMtx = LXMatrix.new()
LMtx.OrthoOffCenterLH(1, 2, 3, 4, 1.0, 1000)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

PerspectiveFovLH

建立左手坐标系下的定制正交透视矩阵

```
PerspectiveFovLH(Fov,Aspect,zn,zf)
```

参数

参数名	类型	描述	是否可选	默认值
Fov	number	视域弧度	否	无
Aspect	number	宽高比例	否	无

zn	number	视区卷最近面的 z 值	否	无
zf	number	视区卷最远面的 z 值	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.PerspectiveFovLH(45, 1, 1, 1000)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

PerspectiveLH

建立左手坐标系下的透视矩阵

```
PerspectiveLH(w,h,zn,zf)
```

参数

参数名	类型	描述	是否可选	默认值
w	number	视区卷宽度	否	无
h	number	视区卷高度	否	无
zn	number	视区卷最近面的 z 值	否	无
zf	number	视区卷最远面的 z 值	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx. PerspectiveLH(1,5, 1,1000)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

PerspectiveOffCenterLH

建立左手坐标系下的定制透视矩阵

```
PerspectiveOffCenterLH (l,r,b,t,zn,zf)
```

参数

参数名	类型	描述	是否可选	默认值
l	number	视区卷最小 X 值	否	无
r	number	视区卷最大 X 值	否	无
b	number	视区卷最小 Y 值	否	无
t	number	视区卷最大 Y 值	否	无
zn	number	视区卷最小 Z 值	否	无
zf	number	视区卷最大 Z 值	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
LMtx = LXMatrix.new()
LMtx.PerspectiveOffCenterLH (1, 2, 3, 4, 1, 1000)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

RotationAxis

建立一个绕任意轴旋转的矩阵

```
RotationAxis(vAxis,Angle)
```

参数

参数名	类型	描述	是否可选	默认值
vAxis	LXVector3	旋转轴向量	否	无
Angle	number	旋转的角度（弧度）	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
vAxis = LXVector3.new(2, 3, 1)
LMtx.RotationAxis(vAxis,PI/2)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

RotationQuaternion

根据一个四元数创建旋转矩阵

```
RotationQuaternion(pQ)
```

参数

参数名	类型	描述	是否可选	默认值
pQ	LXQuaternion	四元数	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
pQ = LXQuaternion.new(1, 2, 1, 1)
LMtx.RotationQuaternion(pQ)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

Rotation

建立绕指定轴旋转矩阵

```
Rotation(Angle， RotateType)
```

参数

参数名	类型	描述	是否可选	默认值
Angle	number	旋转的角度（弧度）	否	无
RotateType	number	旋转的类型	否	无

返回值

```
无
```

注意事项

```
RotateType  旋转的类型取值如下
= 1       绕 X 轴旋转
= 2       绕 Y 轴旋转
= 3       绕 Z 轴旋转
```

示例

```
LMtx = LXMatrix.new()
LMtx.Rotation(PI/2 ， 2)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

Scaling

建立一个缩放矩阵

```
Scaling(sx,sy,sz)
```

参数

参数名	类型	描述	是否可选	默认值
sx	number	X 轴缩放系数	否	无
sy	number	Y 轴缩放系数	否	无
sz	number	Z 轴缩放系数	否	无

返回值

```
无
```

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.Scaling (0.5，0.5，0.5)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

Translation

建立一个位移矩阵

```
Translation (sx,sy,sz)
```

参数

参数名	类型	描述	是否可选	默认值
sx	number	X 轴位移	否	无
sy	number	Y 轴位移	否	无
sz	number	Z 轴位移	否	无

返回值

无

注意事项

无

示例

```
LMtx = LXMatrix.new()
LMtx.Translation (5，15，15)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

TransformVec2Coord

使用当前矩阵转换指定 2 维向量

```
TransformVec2Coord( vcin)
```

参数

参数名	类型	描述	是否可选	默认值
vcin	LXVector2	输入 2 维向量	否	无

返回值

类型	描述
LXVector2	输出 2 维向量

注意事项

（转换过程中先把 pIn 转化为(x,y,0,1)）并调整结果 w 值=1

示例

```
vcin = LXVector2.new(1,1)
LMtx = LXMatrix.new()
Vcout = LMtx.TransformVec2Coord(vcin)
x,y =    Vcout.GetValue()
print(x)
print(y)
```

参见

Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua

TransformVec2Normal

使用当前矩阵转换指定 2 维向量

```
TransformVec2Normal(vcin)
```

参数

参数名	类型	描述	是否可选	默认值
vcin	LXVector2	输入 2 维向量	否	无

返回值

类型	描述
LXVector2	输出 2 维向量

注意事项

（转换过程中先把 pIn 转化为(x,y,0,0)）

示例

```
vcin = LXVector2new(1,1)
LMtx = LXMatrix.new()
Vcout = LMtx.TransformVec2Normal(vcin)
x,y =    Vcout.GetValue()
print(x)
print(y)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

TransformVec3Coord

使用当前矩阵转换指定 3 维向量

```
TransformVec3Coord(vcin)
```

参数

参数名	类型	描述	是否可选	默认值
vcin	LXVector3	输入 3 维向量	否	无

返回值

类型	描述
LXVector3	输出 3 维向量

注意事项

```
（转换过程中先把 pIn 转化为(x,y,z,1)）并调整结果 w 值=1
```

示例

```
vcin = LXVector3.new(1,1,1)
LMtx = LXMatrix.new()
Vcout = LMtx.TransformVec3Coord(vcin)
x,y,z =    Vcout.GetValue()
print(x)
print(y)
Print(z)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

TransformVec3Normal

使用当前矩阵转换指定 3 维向量

```
TransformVec3Normal(vcin)
```

参数

参数名	类型	描述	是否可选	默认值
vcin	LXVector3	输入 3 维向量	否	无

返回值

类型	描述
LXVector3	输出 3 维向量

注意事项

转换过程中先把 pIn 转化为(x,y,z,0)

示例

```
vcin = LXVector3.new(1,1,1)
LMtx = LXMatrix.new()
Vcout = LMtx.TransformVec3Normal(vcin)
x,y,z =    Vcout.GetValue()
print(x)
print(y)
Print(z)
```

参见

```
Samples\Test_LuaBase\Test_LXMatrix\Test_LXMatrix.lua
```

LXPlane

功能描述

该对象实现平面功能。

函数列表

函数名称	说明
CreatePlaneFromPointNormal	根据平面点和法线,构造平面
CreatePlaneFromPoints	根据空间 3 个点,构造平面
ComputerPlaneIntersectLine	计算一条直线和平面的交点
NormalizePlane	规则化平面系数
ScalePlane	根据给定系数缩放平面
TransformPlane	根据给定的矩阵转换平面
GetNormal	获取法线
GetDistance	获取指定点到平面的距离
GetPositionSide	获取与平面的关系
ComputerVector4Dot	计算平面和四维向量的点乘
ComputerVector3Dot	计算平面和 3 维向量的点乘,w 值假设为 1
ComputerVector3DotNormal	计算平面和 3 维向量的点乘,w 值假设为 0

CreatePlaneFromPointNormal

根据平面点和法线,构造平面

```
CreatePlaneFromPointNormal(vec3, normalVec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	平面点	否	无
normalVec3	LXVector3	法线	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new(1, 1, 1)
NormalVec3 = LXVector3.new(2, 3, 4);
Plane = LXPlane.new()
Plane.CreatePlaneFromPointNormal(Vec3, NormalVec3)
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

CreatePlaneFromPoints

根据空间 3 个点,构造平面

```
CreatePlaneFromPoints(vec31, vec32, vec33)
```

参数

参数名	类型	描述	是否可选	默认值
Vec31	LXVector3	顶点 1	否	无
Vec32	LXVector3	顶点 2	否	无
Vec33	LXVector3	顶点 3	否	无

返回值

无

注意事项

无

示例

```
Vec31 = LXVector3.new(1, 1, 1)
Vec32 = LXVector3.new(2, 2, 2)
Vec33 = LXVector3.new(3, 3, 3)
Plane = LXPlane.new()
Plane.CreatePlaneFromPoints(Vec31, Vec32, Vec33)
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

ComputerPlaneIntersectLine

计算一条直线和平面的交点,存在交点返回 TRUE 不存在交点返回 FALSE

```
ComputerPlaneIntersectLine(Vec31, Vec32)
```

参数

参数名	类型	描述	是否可选	默认值
Vec31	LXVector3	直线起点	否	无
Vec32	LXVector3	直线终点	否	无

返回值

类型	描述
boolean	相交返回 true，否则返回 false
LXVector3	相交返回交点，否则返回 nil

注意事项

无

示例

```
Vec31 = LXVector3.new(1, 1, 1)
Vec32 = LXVector3.new(2, 2, 2)
Plane = LXPlane.new()
bIntersect, intersectVec3 = Plane.ComputerPlaneIntersectLine(Vec31, Vec32)
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

NormalizePlane

规则化平面系数

```
NormalizePlane()
```

参数

无

返回值

无

注意事项

无

示例

```
Plane = LXPlane.new()
Plane.NormalizePlane()
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

ScalePlane

根据给定系数缩放平面

```
ScalePlane(fScale)
```

参数

参数名	类型	描述	是否可选	默认值
fScale	number	缩放因子	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new(1, 1, 1)
NormalVec3 = LXVector3.new(2, 3, 4);
Plane = LXPlane.new()
Plane.CreatePlaneFromPointNormal(Vec3, NormalVec3)
Plane.ScalePlane(2.5)
```

参见

```
Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua
```

TransformPlane

根据给定的矩阵转换平面　给定的矩阵必须是实际需要转换矩阵的逆矩阵的转置矩阵

```
TransformPlane(matrix)
```

参数

参数名	类型	描述	是否可选	默认值
Matrix	LXMatrix	变换矩阵	否	无

返回值

无

注意事项

无

示例

```
Matrix = LXMatrix.new()
Matrix.SetRow(1, 1, 2, 3, 4)
Plane = LXPlane.new()
Plane.TransformPlane(matrix)
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

GetNormal

获取法线

```
GetNormal()
```

参数

无

返回值

类型	描述
LXVector3	法线值

注意事项

无

示例

```
Plane = LXPlane.new()
NormalVec3 = Plane.GetNormal()
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

GetDistance

获取指定点到平面的距离

```
GetDistance(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	指点点	否	无

返回值

类型	描述
number	指定点到平面的距离

注意事项

无

示例

```
Vec3 = LXVector3.new(1, 2, 3)
Plane = LXPlane.new()
Dist = Plane.GetDistance(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

GetPositionSide

GetPositionSide(Vec3) 得到指定点在平面的关系
GetPositionSide(centerVec3, radiusVec3) 得到一个包围盒与平面的关系
GetPositionSide(aabbbox) 得到一个包围盒与平面的关系

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	指点点	否	无
centerVec3	LXVector3	包围盒中心点	否	无
radiusVec3	LXVector3	包围盒半径	否	无
Aabbbox	LXAABBBox	包围盒	否	无

返回值

类型	描述
number	与平面的关系

注意事项

```
<0  表示在平面的背面
=0  表示在平面上
>0  表示在平面的正面
```

示例

```
Vec3 = LXVector3.new(2, 3, 5)
radiusVec3 = LXVector3.new(1, 2, 3)
Aabbbox = LXAABBBox.new()
Aabbbox.SetAABBBox(1, 2, 3, 4, 5, 6)
Plane = LXPlane.new()
Side = Plane.GetPositionSide(Vec3)
Side = Plane.GetPositionSide(Vec3, radiusVec3)
Side = Plane.GetPositionSide(aabbbox)
```

参见

```
Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua
```

ComputerVector4Dot

计算平面和四维向量的点乘

```
ComputerVector4Dot(vec4)
```

参数

V 参数名	类型	描述	是否可选	默认值
Vec4	LXVector4	四维向量	否	无

返回值

类型	描述
number	点乘的结果

注意事项

```
无
```

示例

```
Vec4 = LXVector4.new(1, 2, 3, 1)
Plane = LXPlane.new()
```

```
fResult = Plane.ComputerVector4Dot(Vec4)
```

参见

```
Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua
```

ComputerVector3Dot

计算平面和 3 维向量的点乘,w 值假设为 1,该函数主要用来计算维空间点和平面关系

```
ComputerVector3Dot(Vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	3 维向量	否	无

返回值

类型	描述
number	3 维向量的点乘

注意事项

```
无
```

示例

```
Vec3 = LXVector3.new(1, 2, 3)
Plane = LXPlane.new()
fResult = Plane.ComputerVector3Dot(Vec3)
```

参见

```
Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua
```

ComputerVector3DotNormal

计算平面和 3 维向量的点乘,w 值假设为 0,该函数主要用来计算平面法线和其他法线的角度

```
ComputerVector3DotNormal(vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	3 维向量	否	无

返回值

类型	描述
number	3 维向量的点乘

注意事项

无

示例

```
Vec3 = LXVector3.new(1, 2, 3)
Plane = LXPlane.new()
fResult = Plane.ComputerVector3DotNormal(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXPlane\Test_Plane.lua

LXQuaternion

功能描述

本类是对四元数的数学运算及操作的封装类。

函数列表

函数名称	说明
new	新建一个 LXQuaternion 对象
SetValue	设置四元数的值
GetValue	根据四元数旋转一个向量
Rotate	根据以任意轴旋转一个角度构造四元数
FromRotationAxis	根据以任意轴旋转一个角度构造四元数
FromOrthoAxis	根据给定的 3 个正交的标准轴构造四元数
FromRotationMatrix	根据一个旋转矩阵构造四元数
FromRotationYawPitchRoll	根据给定的欧拉角构造四元数
FromPoints	计算从一个方向变换到另一个方向的四元数
ToAxisAngle	计算一个四元数的旋转轴和旋转方向

ToOrthoAxis	计算一个四元数对应的个正交的标准轴
ConvertConjugate	得到指定四元数的共轭四元数
DotProduct	计算两个四元数的点乘
Exp	计算四元数的幂
Ln	计算一个四元数的对数四元数
SetIdentity	把四元数转换为特性四元数
Inverse	得到四元数的逆
UnitInverse	得到单位四元数的逆
IsIdentity	四元数是否为特性四元数
IsUnit	四元数是否为单位四元数
GetLength	得到四元数长度
Multiple	计算两个四元数的乘积
Normalize	规则化指定四元数
Slerp	计算两个四元数球面线性插值
rSquadSlerp	计算两个四元数球面四边形插值
SetupSquadSlerp	建造球面四边形插值控制点

new

创建一个 LXQuaternion 对象

```
new (x,y,z,w)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	x 分量值	否	无
y	number	y 分量值	否	无
z	number	z 分量值	否	无
w	number	w 分量值	否	无

返回值

无

注意事项

无

示例

```
LQn = LXQuaternion.new(5，6，7，10)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

SetValue

设置四元数的值

SetValue(x,y,z,w)

参数

参数名	类型	描述	是否可选	默认值
x	number	x 分量值	否	无
y	number	y 分量值	否	无
z	number	z 分量值	否	无
w	number	w 分量值	否	无

返回值

无

注意事项

无

示例

```
LQn = LXQuaternion.new()
LQn.SetValue(5，6，7，10)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

GetValue

获取四元数的值

GetValue()

参数

无

返回值

类型	描述
number	x 分量值

number	y 分量值
number	z 分量值
number	w 分量值

注意事项

无

示例

```
x,y,z,w = LQn.GetValue()
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

Rotate

根据四元数旋转一个向量

```
Rotate (vec3)
```

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	要旋转的向量	否	无

返回值

类型	描述
LXVector3	旋转后的新向量

注意事项

无

示例

```
Vec3 = LXVector3.new(1, 2, 1)
Vec3 = LQn.Rotate(Vec3)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

FromRotationAxis

根据任意轴旋转一个角度构造四元数

```
FromRotationAxis(Vec3,Angle)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3	LXVector3	旋转的向量	否	无
Angle	number	旋转的角度	否	无

返回值

无

注意事项

无

示例

```
Vec3 = LXVector3.new(2, 1, 0)
LQn.FromRotationAxis(Vec3, 30)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

FromOrthoAxis

根据给定的 3 个正交的标准轴构造四元数

```
FromOrthoAxis(pV1,pV2,pV3)
```

参数

参数名	类型	描述	是否可选	默认值
pV1	LXVector3	正交轴	否	无
pV2	LXVector3	正交轴	否	无
pV3	LXVector3	正交轴	否	无

返回值

无

注意事项

无

示例

```
Vec1 = LXVector3.new(1, 0, 0)
Vec2 = LXVector3.new(0, 1, 0)
Vec3 = LXVector3.new(0, 0, 1)
LQn. FromOrthoAxis(Vec1, Vec2, Vec3)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

FromRotationMatrix

根据一个旋转矩阵构造四元数

```
FromRotationMatrix(mtx)
```

参数

参数名	类型	描述	是否可选	默认值
mtx	LXMatrix	矩阵	否	无

返回值

无

注意事项

无

示例

```
mtx = LXMatrix.new()
mtx.SetRow(1, 1, 0, 1, 1)
LQn.FromRotationMatrix(mtx)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

FromRotationYawPitchRoll

根据给定的欧拉角构造四元数

```
FromRotationYawPitchRoll (Yaw,Pitch,Roll)
```

参数

参数名	类型	描述	是否可选	默认值
Yaw	number	围绕 Y 轴旋转	否	无
Pitch	number	围绕 X 轴旋转	否	无
Roll	number	围绕 Z 轴旋转	否	无

返回值

无

注意事项

无

示例

```
LQn.FromRotationYawPitchRoll(PI/2,PI/3,PI/6)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

FromPoints

计算从一个方向变换到另一个方向的四元数

```
FromPoints (Vec3From, Vec3To)
```

参数

参数名	类型	描述	是否可选	默认值
Vec3From	LXVector3	起点	否	无
Vec3To	LXVector3	终点	否	无

返回值

无

注意事项

无

示例

```
Vec3From = LXVector3.new(1, 0, 1)
Vec3To = LXVector3.new(0, 4, 2)
```

```
LQn. FromPoints (Vec3From, Vec3To)
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

ToAxisAngle

计算一个四元数的旋转轴和旋转方向

```
ToAxisAngle()
```

参数

```
无
```

返回值

类型	描述
LXVector3	旋转向量
number	旋转角度(以弧度为单位)

注意事项

```
无
```

示例

```
vcOut, fAngle = LQn.ToAxisAngle()
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

ToOrthoAxis

计算一个四元数对应的 3 个正交的标准轴

```
ToOrthoAxis()
```

参数

```
无
```

返回值

类型	描述
LXVector3	正交轴
LXVector3	正交轴

LXVector3	正交轴

注意事项

无

示例

```
Vec1,Vec2,Vec3 = LQn.ToOrthoAxis()
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

ConvertConjugate

得到指定四元数的共轭四元数

```
ConvertConjugate()
```

参数

无

返回值

无

注意事项

结果赋给自身

示例

```
LQn.ConvertConjugate()
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

DotProduct

计算两个四元数的点乘

```
DotProduct(Qtn)
```

参数

参数名	类型	描述	是否可选	默认值

Qtn	LXQuaternion	四元数	否	无

返回值

类型	描述
number	点乘值

注意事项

结果赋给自身

示例

```
Qtn = LXQuaternion.new(2, 0, 1, 1)
fResult = LQn.DotProduct(Qtn)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

Exp

计算四元数的幂

```
Exp()
```

参数

无

返回值

无

注意事项

该函数转换一个纯四元数(q = (0, theta * v),w 忽略)到一个单位四元数(exp(Q) = (cos(theta), sin(theta) * v) |v| = 1)
结果赋给自身

示例

```
LQn.Exp()
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

Ln

计算一个四元数的对数四元数

```
Ln(Qtn)
```

参数

参数名	类型	描述	是否可选	默认值
Qtn	LXQuaternion	四元数	否	无

返回值

无

注意事项

该函数转换一个单位四元数(Q=(cos(theta),sin(theta)*v) |v| = 1)到纯四元数(q = (0, theta * v),w 忽略)

示例

```
Qtn = LXQuaternion.new(1, 0, 0, 1)
LQn.Ln(Qtn)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

SetIdentity

把四元数转换为特性四元数(x,y,z,w)-->(0,0,0,1)

```
SetIdentity()
```

参数

无

返回值

无

注意事项

无

示例

```
LQn.SetIdentity()
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

Inverse

得到四元数的逆

```
Inverse()
```

参数

```
无
```

返回值

类型	描述
LXQuaternion	逆四元数

注意事项

```
无
```

示例

```
InverseQn = LQn.Inverse()
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

UnitInverse

得到单位四元数的逆

```
UnitInverse()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
unitInverQtn = LQn.UnitInverse()
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

IsIdentity

四元数是否为特性四元数

```
IsIdentity()
```

参数

```
无
```

返回值

类型	描述
Boolean	若为特性四元数返回 true，否则返回 false

注意事项

```
无
```

示例

```
bResult = LQn.IsIdentity()
if bResult == true then
        print("该四元数是特性四元数")
else
        print("该四元数不是特性四元数")
end
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

IsUnit

四元数是否为单位四元数

```
IsUnit()
```

参数

```
无
```

返回值

类型	描述
Boolean	若为单位四元数返回 true，否则返回 false

注意事项

无

示例

```
bResult = LQn.IsUnit()
if bResult == true then
    print("该四元数是单位四元数")
else
     print("该四元数不是单位四元数")
end
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

GetLength

得到四元数长度

```
GetLength()
```

参数

无

返回值

类型	描述
number	四元数的长度

注意事项

无

示例

```
fLength = LQn. GetLength ()
print("该四元数长度为:")
print(fLength)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

Multiple

计算两个四元数的乘积

```
Multiple(Qtn1,Qtn2)
```

参数

参数名	类型	描述	是否可选	默认值
Qtn1	LXQuaternion	四元数	否	无
Qtn2	LXQuaternion	四元数	否	无

返回值

无

注意事项

乘积赋给自身

示例

```
Qtn1 = LXQuaternion.new(1, 1, 0, 1)
Qtn2 = LXQuaternion.new(0, 2, 2, 1)
LQn. Multiple(Qtn1,Qtn2)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

Normalize

规则化指定四元数

```
Normalize()
```

参数

无

返回值

无

注意事项

无

示例

```
LQn.Normalize()
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

Slerp

计算两个四元数球面线性插值

```
Slerp(Qtn1,Qtn2,f)
```

参数

参数名	类型	描述	是否可选	默认值
Qtn1	LXQuaternion	四元数	否	无
Qtn2	LXQuaternion	四元数	否	无
f	number	增加值	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Qtn1 = LXQuaternion.new(1, 1, 0, 1)
Qtn2 = LXQuaternion.new(0, 2, 2, 1)
LQn. Slerp(Qtn1,Qtn2,0.1)
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

SquadSlerp

计算两个四元数球面四边形插值

```
SquadSlerp(Qtn1,QtnA,QtnB,QtnC,f)
```

参数

参数名	类型	描述	是否可选	默认值
Qtn1	LXQuaternion	四元数	否	无
QtnA	LXQuaternion	四元数	否	无
QtnB	LXQuaternion	四元数	否	无

QtnC	LXQuaternion	四元数	否	无
f	number	增加值	否	无

返回值

无

注意事项

该函数使用一系列球面线性插值实现

示例

```
LQn.SquadSlerp(Qtn1,pA,pB,pC,0.1)
```

参见

Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua

SetupSquadSlerp

建造球面四边形插值控制点

```
SetupSquadSlerp(Qtn0,Qtn1,Qtn2,Qtn3)
```

参数

参数名	类型	描述	是否可选	默认值
Qtn0	LXQuaternion	四元数	否	无
Qtn1	LXQuaternion	四元数	否	无
Qtn2	LXQuaternion	四元数	否	无
Qtn3	LXQuaternion	四元数	否	无

返回值

类型	描述
QtnOut1	四元素
QtnOut2	四元数
QtnOut3	四元数

注意事项

无

示例

```
Qn1 = LXQuaternion.new(1,2,3,PI/2)
QnA = LXQuaternion.new(4,5,6,PI/3)
QnB = LXQuaternion.new(7,8,9,PI/4)
QnC = LXQuaternion.new(10,11,12,PI/6)
AOut,BOut,COut = LQn.SetupSquadSlerp(Qn1, QnA, QnB, QnC)
```

参见

```
Samples\Test_LuaBase\Test_LXQuaternion\Test_LXQuaternion.lua
```

LXVector2

功能描述

该对象实现二维向量的数学运算及操作。

属性列表

属性名称	说明
x	LXVector2 的 x 属性
y	LXVector2 的 y 属性

函数列表

函数名称	说明
new	创建一个 LXVector2 对象
GetValue	得到二维向量的值
SetValue	设置二维向量的值
“+”运算符	两个向量相加
“-”运算符	两个向量相减
“*”运算符	缩放向量
“/”运算符	缩放向量
“==” 运算符	判断两个向量是否绝对相等
Invert	设置 2 维向量各分量值为其倒数
Negate	设置 2 维向量各分量值为其负数
IsSame	该向量是否与指定向量近似相等
Normalize	规则化 2 维向量
GetLength	得到 2 维向量长度
Scale	缩放 2 维向量

Distance	计算两个 2 维向量距离
DotProduct	计算两个 2 维向量点乘
CrossProduct	计算两个 2 维向量叉乘的 z 分量 (x,y,0) pV1->x * pV2->y - pV1->y * pV2->x 如果 z 分量>0.0f 说明该向量与本向量是逆时针方向
GetAngle	无参数：得到指定向量的角度,逆时针方向 有参数：得到与指定向量的夹角
SetLength	设置向量的长度
Lerp	执行线性插值
Rotate	绕指定中心点旋转一个角度
Reflect	计算基于法平面的反射向量
Refract	1 计算基于法平面的反射向量 2 计算基于法平面的折射向量
Project	计算向量在法线上的投影
SetX	设置 X 值
SetY	设置 Y 值
GetX	得到 X 值
GetY	得到 Y 值
Absolute	分量取绝对值

new

创建一个 LXVector2 对象

```
new(Vector1)
new(x , y)
```

参数

方法 1:

参数名	类型	描述	是否可选	默认值
Vector1	LXVector2	LXVector2 对象	是	(0,0)

方法 2:

参数名	类型	描述	是否可选	默认值
x	number	二维向量 x 值	是	0
y	number	二维向量 y 值	是	0

返回值

类型	描述
LXVector2	表示要得到的值

注意事项

无

示例

```
Vector1 = LXVector2.new()
Vector2 = LXVector2.new(Vector1)
Vector3 = LXVector2.new(50.0,60.0)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

GetValue

得到二维向量值

```
GetValue()
```

参数

无

返回值

类型	描述
number	二维向量 x 值
number	二维向量 y 值

注意事项

无

示例

```
x , y = Vector1.GetValue()
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

SetValue

设置二维向量的值

```
SetValue(x , y)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	二维向量 x 值	否	无
y	number	二维向量 y 值	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vector1.SetValue(50 , 90)
```

参见

```
Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua
```

“+”运算符

两个向量相加

```
Vector3 = Vector1 + Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector 1	LXVector2 对象	操作数	否	无
Vector 2	LXVector2 对象	操作数	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vector3 = Vector1 + Vector2
```

参见

```
Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua
```

“-”运算符

两个向量相减

```
Vector3 = Vector1 - Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector 1	LXVector2 对象	操作数	否	无
Vector 2	LXVector2 对象	操作数	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vector3 = Vector1 - Vector2
```

参见

```
Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua
```

“*”运算符

缩放向量

```
Vector3 = Vector1 *f
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector2 对象	操作数	否	无
f	number	操作数	否	无

返回值

无

注意事项

无

示例

```
Vector3 = Vector1 *0.1
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

“/”运算符

缩放向量

```
Vector3 = Vector1 /f
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector2 对象	操作数	否	无
f	number	操作数	否	无

返回值

无

注意事项

无

示例

```
Vector3 = Vector1 /10
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

“==”运算符

判断两个向量是否绝对相等

```
B = Vector3 == Vector1
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector2 对象	操作数	否	无
Vector3	LXVector2 对象	操作数	否	无

返回值

类型	描述
boolean	true 表肯定， false 表否定

注意事项

无

示例

```
B = Vector3 == Vector1
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Invert

设置 2 维向量各分量值为其倒数

```
Invert()
```

参数

无

返回值

无

注意事项

示例

```
Vector1.Invert()
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Negate

设置 2 维向量各分量值为其负数

```
Negate()
```

参数

无

返回值

无

注意事项

无

示例

```
Vector1. Negate()
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

IsSame

该向量是否与指定向量近似相等

```
IsSame(Vector)
```

参数

参数名	类型	描述	是否可选	默认值
Vector	number	LXVector2 对象	否	无

返回值

类型	描述
Boolean	true 表示相等，false 表示不相等

注意事项

无

示例

```
vec21 = LXVector2.new()
bSame = Vector1.IsSame(vec21)
print(bSame)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Normalize

规则化 2 维向量

```
Normalize()
```

参数

无

返回值

无

注意事项

无

示例

```
Vector1.Normalize()
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

GetLength

得到 2 维向量长度

```
GetLength()
```

参数

无

返回值

类型	描述
number	表示要得到的值

注意事项

无

示例

```
Len = Vector1.GetLength()
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Scale

缩放 2 维向量

```
Scale(fScale)
```

参数

参数名	类型	描述	是否可选	默认值
fScale	number	缩放因子	否	无

返回值

无

注意事项

无

示例

```
Vector1.Scalek(0.5)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Distance

计算两个 2 维向量距离

```
Distance(vectorSrc)
```

参数

参数名	类型	描述	是否可选	默认值
vectorSrc	LXVector2	LXVector2 对象	否	无

返回值

类型	描述
number	表示要得到的值

注意事项

无

示例

```
Len = Vector1.Distance(vector1)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

DotProduct

计算两个 2 维向量点乘

```
DotProduct (vector)
```

参数

参数名	类型	描述	是否可选	默认值
vector	LXVector2	LXVector2 对象	否	无

返回值

类型	描述
number	表示要得到的值

注意事项

无

示例

```
dot = Vector1. DotProduct (Vector2)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

CrossProduct

计算两个 2 维向量叉乘的 z 分量　(x,y,0)　　pV1->x * pV2->y - pV1->y * pV2->x
如果 z 分量>0.0f 说明该向量与本向量是逆时针方向

```
CrossProduct (vector)
```

参数

参数名	类型	描述	是否可选	默认值
vector	LXVector2	LXVector2 对象	否	无

返回值

类型	描述
number	表示本二维向量和 vector 向量叉乘的结果

注意事项

无

示例

```
cross = Vector1. CrossProduct (Vector2)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Rotate

绕指定中心点旋转一个角度

```
Rotate (fAngle , vectorCenter)
```

参数

参数名	类型	描述	是否可选	默认值
fAngle	number	旋转的弧度	否	无
vectorCenter	LXVector2	围绕的中心点	否	无

返回值

无

注意事项

无

示例

```
Vector1.Rotate (2.01, Vector2)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

GetAngle

方法 1:得到指定向量的角度,逆时针方向　0-2PI
方法 2:得到与指定向量的夹角　0-2PI

```
GetAngle ()
GetAngle (vector)
```

参数

参数名	类型	描述	是否可选	默认值
vector	LXVector2	LXVector2 对象	否	无

返回值

类型	描述
number	夹角的弧度 值为 0 - 2PI 之间

注意事项

无

示例

```
fAngle = Vector1. GetAngle (Vector2)
fAngle = Vector1. GetAngle ()
```

参见

```
Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua
```

SetLength

设置向量的长度

```
SetLength(fLen)
```

参数

参数名	类型	描述	是否可选	默认值
fLen	number	向量长度	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vector1.SetLength (50.0)
```

参见

```
Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua
```

Lerp

执行线性插值

```
Lerp (Vector1, f)
Lerp(Vector1, Vector2, f)
```

参数

方法 1:

参数名	类型	描述	是否可选	默认值
Vector1	LXVector2	LXVector2 对象	否	无
f	number	LXVector2 对象	否	无

方法 2:

参数名	类型	描述	是否可选	默认值
Vector1	LXVector2	LXVector2 对象	否	无
Vector2	LXVector2	LXVector2 对象	否	无
f	number	倍值	否	无

返回值

无

注意事项

无

示例

```
Vector1.Lerp (Vector2,50.0)
Vector1.Lerp (Vector2, Vector3, 50.0)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Reflect

计算基于法平面的反射向量

```
Reflect (vcNormal)
Reflect(vcSource, vcNormal)
```

参数

参数名	类型	描述	是否可选	默认值
vcSource	LXVector2	LXVector2 对象	否	无
vcNormal	LXVector2	LXVector2 对象	否	无

返回值

无

注意事项

无

示例

```
Vector1.Reflect (Vector2)
Vector1.Reflect (Vector2, Vector3)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Refract

计算基于法平面的反射向量

```
Refract (vcNormal, fIncident, fTransmitted)
Refract(vcSource, vcNormal, fIncident, fTransmitted)
```

参数

参数名	类型	描述	是否可选	默认值
vcSource	LXVector2	LXVector2 对象	否	无
vcNormal	LXVector2	LXVector2 对象	否	无
fIncident	number	表示折射率	否	无
fTransmitted	number	表示折射率	否	无

返回值

无

注意事项

无

示例

```
Vector1.Refract (Vector2, 20.0,30.0)
Vector1.Refract (Vector2, Vector3, 20.0,30.0)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Project

计算向量在法线上的投影

```
Project (vcNormal)
Project (vcSource, vcNormal)
```

参数

参数名	类型	描述	是否可选	默认值
vcSource	LXVector2	LXVector2 对象	否	无
vcNormal	LXVector2	LXVector2 对象	否	无

返回值

无

注意事项

无

示例

```
Vector1.Project (Vector2)
Vector1.Project (Vector2, Vector3)
```

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

SetX

设置 X 值

```
SetX（nVal）
```

参数

参数名	类型	描述	是否可选	默认值
nVal	number	要设置的值	否	无

返回值

无

注意事项

无

示例

```
Vector1. SetX (80)
```

参见

```
Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua
```

GetX

得到 X 值

```
GetX（）
```

参数

```
nVal: number 类型  表示要得到的值
```

返回值

类型	描述
number	表示要得到的值

注意事项

```
无
```

示例

```
n = Vector1.GetX ()
```

参见

```
Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua
```

SetY

设置 Y 值

```
SetY（nVal）
```

参数

参数名	类型	描述	是否可选	默认值
nVal	number	要设置的值	否	无

返回值

无

注意事项

无

示例

Vector1. SetY (80)

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

GetY

得到 Y 值

GetY（）

参数

无

返回值

类型	描述
number	表示要得到的值

注意事项

无

示例

n = Vector1.GetY ()

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

Absolute

分量取绝对值

Absolute ()

参数

无

返回值

无

注意事项

无

示例

Vector1.Absolute ()

参见

Samples\Test_LuaBase\Test_LXVector2\脚本\XMain.lua

LXVector3

功能描述

该对象实现三维向量的数学运算及操作。

属性列表

属性名称	说明
x	LXVector3 的 x 属性
y	LXVector3 的 y 属性
z	LXVector3 的 z 属性

函数列表

函数名称	说明
new	创建一个 LXVector3 对象
SetValue	设置值
GetValue	获取值
“+”运算符	两个向量相加

“-”运算符	两个向量相减
“*”运算符	缩放向量
“/”运算符	缩放向量
“==” 运算符	判断两个向量是否绝对相等
Absolute	分量取绝对值
Invert	分量取倒数
Negate	分量取负数
IsDirSame	判断两向量方向是否近似相等
Normalize	规则化向量
Length	得到向量长度
Scale	缩放向量
Distance	计算两向量距离
DotProduct	计算两向量点乘
CrossProduct	计算两向量叉乘
IsBetweenPoints	判断点是否在指定两点确定的线段上
SetLength	设置向量的长度
Rotate	绕指定中心点旋转一个角度
Lerp	执行线性插值,结果赋给自己
ComputeHeadingAndPitch	计算向量在 XZ 平面上的投影角度和 XY 平面上投影角度
SetFromHeadingAndPitch	通过投影角度构造向量
SetFromSphericalCoords	通过球体经纬度构造向量
Reflect	计算基于法平面的反射向量
Refract	计算基于法平面的折射向量
Project	计算向量在法线上的投影
CreateNormal	根据给定的顺时针点,或者根据给定的两个向量，计算该平面法线

new

创建一个 LXVector3 对象

```
new ()
new(Vec)
new(x, y, z)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	三维向量 x 值	是	0
y	number	三维向量 y 值	是	0
z	number	三维向量 z 值	是	0

参数名	类型	描述	是否可选	默认值
Vec	LXVector3	三维向量	否	无

返回值

类型	描述
LXVector3	LXVector3 对象

注意事项

无

示例

```
Vec = LXVector3.new()      -- x = 0 y = 0 z = 0
Vec1 = LXVector3.new (-2,3,-4)      --x = -2 y = 3 z = -4
Vec2 = LXVector3.new(Vec1)         --Vec2 = Vec1
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

SetValue

设置向量各分量值

```
SetValue(x,y,z)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	三维向量 x 值	是	0
y	number	三维向量 y 值	是	0
z	number	三维向量 z 值	是	0

返回值

无

注意事项

无

示例

```
Vec = LXVector3.new()        -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)        --x = -2 y = 3 z = -4
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

GetValue

获取向量各分量值

```
GetValue()
```

参数

无

返回值

类型	描述
number	X 分量值
number	Y 分量值
number	Z 分量值

注意事项

无

示例

```
Vec = LXVector3.new()        -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)        --x = -2 y = 3 z = -4
x,y,z = Vec.GetValue()
print(x)            --输出-2
print(y)            --输出 3
print(z)            --输出-4
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

“+”运算符

两个向量相加

```
Vector3 = Vector1 + Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector 1	LXVector3 对象	操作数	否	无
Vector 2	LXVector3 对象	操作数	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vector3 = Vector1 + Vector2
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

“-”运算符

两个向量相减

```
Vector3 = Vector1 - Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector 1	LXVector3 对象	操作数	否	无
Vector 2	LXVector3 对象	操作数	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vector3 = Vector1 - Vector2
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

“*”运算符

向量的乘运算

```
Vector3 = Vector1 *f
Vector3 = Vector1 * Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector3 对象	操作数	否	无
Vector2	LXVector3 对象	操作数	否	无
f	number	操作数	否	无

返回值

无

注意事项

若两个操作数都是向量时，调用 CrossProduct 函数

示例

```
Vector3 = Vector1 *0.1
Vector3 = Vector1 * Vector2
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

“/”运算符

缩放向量

```
Vector3 = Vector1 /f
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector3 对象	操作数	否	无
f	number	操作数	否	无

返回值

无

注意事项

无

示例

```
Vector3 = Vector1 /10
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

“==”运算符

判断两个向量是否绝对相等

```
B = Vector3 == Vector1
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector3 对象	操作数	否	无
Vector3	LXVector3 对象	操作数	否	无

返回值

类型	描述
boolean	true 表肯定， false 表否定

注意事项

无

示例

```
B = Vector3 == Vector1
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Absolute

分量取绝对值

```
Absolute()
```

参数

无

返回值

无

注意事项

无

示例

```
Vec = LXVector3.new()          -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)          --x = -2 y = 3 z = -4
Vec.Absolute()
x,y,z = Vec.GetValue()
print(x)                --输出 2
print(y)                --输出 3
print(z)                --输出 4
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Invert

分量取倒数

```
Invert()
```

参数

无

返回值

无

注意事项

无

示例

```
Vec = LXVector3.new()        -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)        --x = -2 y = 3 z = -4
Vec.Invert()
x,y,z = Vec.GetValue()
print(x)              --输出-0.5
print(y)              --输出 0.333333
print(z)              --输出-0.25
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

Negate

分量取负数

```
Negate()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
Vec = LXVector3.new()        -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)        --x = -2 y = 3 z = -4
Vec.Negate()
x,y,z = Vec.GetValue()
print(x)              --输出 2
print(y)              --输出-3
print(z)              --输出 4
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

IsDirSame

判断两向量方向是否近似相等

```
IsDirSame(Vec,Error)
```

参数

参数名	类型	描述	是否可选	默认值
Vec	LXVector3	指定向量	否	无
Error	number	容差	否	无

返回值

类型	描述
Boolean	是否相等

注意事项

无

示例

```
Vec = LXVector3.new()      -- x = 0 y = 0 z = 0
Vec1 = LXVector3.new (-2,3,-4)      --x = -2 y = 3 z = -4
res = Vec.IsDirSame(Vec1, 0.1)
print(res)                --输出 false
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Normalize

规则化向量

```
Normalize ()
```

参数

无

返回值

无

注意事项

无

示例

```
Vec = LXVector3.new()        -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)        --x = -2 y = 3 z = -4
Vec.Normalize()
x,y,z = Vec.GetValue()
print(x)              --输出-0.371391
print(y)              --输出 0.557086
print(z)              --输出-0.742781
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Length

得到向量长度

```
Length ()
```

参数

无

返回值

类型	描述
number	向量长度

注意事项

无

示例

```
Vec = LXVector3.new()        -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)        --x = -2 y = 3 z = -4
res = Vec.Length()
print(res)              --输出 5.385165
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Scale

缩放向量

```
Scale (var)
```

参数

参数名	类型	描述	是否可选	默认值
var	number	缩放因子	否	无

返回值

无

注意事项

无

示例

```
Vec = LXVector3.new()        -- x = 0 y = 0 z = 0
Vec.SetValue(-2,3,-4)        --x = -2 y = 3 z = -4
Vec.Sclae(0.1)
x,y,z = Vec.GetValue()
print(x)              --输出-0.2
print(y)              --输出 0.3
print(z)              --输出-0.4
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Distance

计算两向量距离

```
Distance (Vec)
```

参数

参数名	类型	描述	是否可选	默认值
Vec	LXVector3	指定向量	否	无

返回值

类型	描述
number	两向量之间的距离

注意事项

无

示例

```
Vec = LXVector3.new()      -- x = 0 y = 0 z = 0
Vec1 = LXVector3.new (-2,3,-4)      --x = -2 y = 3 z = -4
res = Vec.Distance(Vec1)
print(res)              --输出 5.385165
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

DotProduct

计算两向量点乘

```
DotProduct(Vec)
```

参数

参数名	类型	描述	是否可选	默认值
Vec	LXVector3	指定向量	否	无

返回值

类型	描述
number	两向量点乘结果

注意事项

无

示例

```
Vec = LXVector3.new()      -- x = 0 y = 0 z = 0
Vec1 = LXVector3.new (-2,3,-4)      --x = -2 y = 3 z = -4
```

```
res = Vec.DotProduct(Vec1)
print(res)              --输出 0
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

CrossProduct

计算两向量叉乘

```
CrossProduct(Vec1)
CrossProduct(Vec1, Vec2)
```

参数

参数名	类型	描述	是否可选	默认值
Vec1	LXVector3	指定向量	否	无

参数名	类型	描述	是否可选	默认值
Vec1	LXVector3	指定向量	否	无
Vec2	LXVector3	指定向量	否	无

返回值

```
无
```

注意事项

```
对于形式一：计算本身与 Vec1 的叉乘，把结果保存在自身
对于形式二：计算 Vec1 与 Vec2 的叉乘，把结果保存在自身
```

示例

```
--形式一
Vec = LXVector3.new(1, 2, 2)
Vec1 = LXVector3.new(1, 2, 3)
Vec.CrossProduct(Vec1)
x,y,z = Vec.GetValue()
print(x)              --输出 2
print(y)              --输出-1
print(z)              --输出 0
--形式二
Vec = LXVector3.new()
Vec1 = LXVector3.new(1, 2, 2)
```

```
Vec2 = LXVector3.new(1, 2, 3)
Vec.CrossProduct(Vec1, Vec2)
x,y,z = Vec.GetValue()
print(x)            --输出 2
print(y)            --输出-1
print(z)            --输出 0
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

IsBetweenPoints

判断点是否在指定两点确定的线段上

```
IsBetweenPoints(VecStart, VecEnd)
```

参数

参数名	类型	描述	是否可选	默认值
VecStart	LXVector3	指定起始坐标点	否	无
VecEnd	LXVector3	指定结束坐标点	否	无

返回值

类型	描述
Boolean	true 表肯定，false 表否定

注意事项

```
无
```

示例

```
Vec = LXVector3.new()
VecStart = LXVector3.new(1, 2, 2)
VecEnd = LXVector3.new(1, 2, 3)
res = Vec.IsBetweenPoints(VecStart, VecEnd)
print(res)          --输出 false
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

SetLength

设置向量的长度

```
SetLength(fLength)
```

参数

参数名	类型	描述	是否可选	默认值
fLength	number	向量指定长度	否	无

返回值

无

注意事项

无

示例

```
Vec.SetLength(10.0)        --设置向量长度为 10
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Rotate

绕指定中心点旋转一个角度

```
Rotate(Angle,Vec,Type)
```

参数

参数名	类型	描述	是否可选	默认值
Angle	number	旋转的角度	否	无
Vec	LXVector3	旋转绕的中心点	否	无
Type	string	旋转类型	否	无

返回值

无

注意事项

Type 取值如下:
“X” 或”x”: 绕 X 轴旋转
“Y” 或”y”: 绕 Y 轴旋转
“Z” 或”z”: 绕 Z 轴旋转

示例

```
Vec = LXVector3.new(1, 2, 2)
Vec1 = LXVector3.new(1, 2, 3)
Vec.Rotate(90, Vec1, “x”)
x,y,z = Vec.GetValue()
print(x)            --输出 1
print(y)            --输出 2.893997
print(z)            --输出 3.448074
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

Lerp

执行线性插值

```
Lerp(Vec1, f)
Lerp(Vec1,Vec2, f)
```

参数

参数名	类型	描述	是否可选	默认值
Vec1	LXVector3	指定向量	否	无
f	number	插值因子	否	无

参数名	类型	描述	是否可选	默认值
Vec1	LXVector3	指定向量 1	否	无
Vec2	LXVector3	指定向量 2	否	无
f	number	插值因子	否	无

返回值

```
无
```

注意事项

```
形式一：自身对象与 Vec1 对象之间插值，结果赋给自身
形式二：Vec1 对象与 Vec 对象之间插值，结果赋给自身
```

示例

```
--形式一
Vec = LXVector3.new(1, 2, 2)
Vec1 = LXVector3.new(1, 2, 3)
Vec.Lerp(Vec1, 1.0)
x,y,z = Vec.GetValue()
print(x)            --输出 1
print(y)            --输出 2
print(z)            --输出 3
--形式二
Vec = LXVector3.new()
Vec1 = LXVector3.new(1, 2, 2)
Vec2 = LXVector3.new(1, 2, 3)
Vec.Lerp(Vec1, Vec2)
x,y,z = Vec.GetValue()
print(x)            --输出 0
print(y)            --输出 0
print(z)            --输出 0
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

ComputeHeadingAndPitch

计算向量在 XZ 平面上的投影角度和 XY 平面上投影角度

```
ComputeHeadingAndPitch()
```

参数

无

返回值

类型	描述
number	在 XZ 平面上的投影角度
number	在 XY 平面上的投影角度

注意事项

无

示例

```
Vec = LXVector3.new(1, 1, 1)
```

```
heading , pitch    = Vec.ComputeHeadingAndPitch()
print(heading)        --输出-0.785398
print(pitch)          --输出 1.5707
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

SetFromHeadingAndPitch

通过投影角度构造向量

```
SetFromHeadingAndPitch (heading, pitch)
```

参数

参数名	类型	描述	是否可选	默认值
heading	number	XZ 平面上的投影角度	否	无
pitch	number	XY 平面上的投影角度	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vec = LXVector3.new()
Vec.SetFromHeadingAndPitch (-0.78, 1.57)
x,y,z = Vec.GetValue()
print(x)          --输出 0
print(y)          --输出 0.2
print(z)          --输出 0.9
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

SetFromSphericalCoords

通过球体经纬度构造向量

```
SetFromSphericalCoords (longitude, latitude)
```

参数

参数名	类型	描述	是否可选	默认值
longitude	number	球体经度	否	无
latitude	number	球体纬度	否	无

返回值

无

注意事项

无

示例

```
Vec = LXVector3.new()
Vec.SetFromHeadingAndPitch (10, 20)
x,y,z = Vec.GetValue()
print(x)            --输出 0
print(y)            --输出 0
print(z)            --输出 1
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Reflect

计算基于法平面的反射向量

```
Reflect(VecNormal)
```

参数

参数名	类型	描述	是否可选	默认值
VecNormal	LXVector3	反射面法向量	否	无

返回值

无

注意事项

求取自身做为入射向量相对反射面的反射向量，结果赋给自身

示例

```
Vec = LXVector3.new(1, 2, 3)
VecNormal = LXVector3.new(1, 1, 1)
Vec.Reflect (VecNormal)
x,y,z = Vec.GetValue()
print(x)            --输出 11
print(y)            --输出 10
print(z)            --输出 9
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

Refract

计算基于法平面的折射向量

```
Refract (VecNormal, )
```

参数

参数名	类型	描述	是否可选	默认值
VecNormal	LXVector3	反射面法向量	否	无
Incident	number	入射方介质参数	否	无
Transmit	number	折射方介质参数	否	无

返回值

无

注意事项

求取自身做为入射向量通过折射面的折射向量，结果赋给自身
折射率 n = Incident/Transmit

示例

```
Vec = LXVector3.new(1, 2, 3)
VecNormal = LXVector3.new(1, 1, 1)
Vec.Refract(VecNormal, 1.0, 0.5)
x,y,z = Vec.GetValue()
print(x)            --输出 2.12
print(y)            --输出 4.12
```

```
print(z)		--输出 6.12
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

Project

计算向量在法线上的投影

```
Project (VecNormal)
```

参数

参数名	类型	描述	是否可选	默认值
VecNormal	LXVector3	指定法向量	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vec = LXVector3.new(1, 2, 3)
VecNormal = LXVector3.new(1, 1, 1)
Vec.Project(VecNormal)
x,y,z = Vec.GetValue()
print(x)		--输出 6
print(y)		--输出 6
print(z)		--输出 6
```

参见

```
Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua
```

CreateNormal

根据给定的顺时针点，或者根据给定的两个向量计算该平面法线

```
形式一：CreateNormal(Vec1,Vec2,Vec3)
形式二：CreateNormal(Vec1,Vec2)
```

参数

参数名	类型	描述	是否可选	默认值
Vec1	LXVector3	指定点 1	否	无
Vec2	LXVector3	指定点 2	否	无
Vec3	LXVector3	指定点 3	否	无

参数名	类型	描述	是否可选	默认值
Vec1	LXVector3	指定向量 1	否	无
Vec2	LXVector3	指定向量 2	否	无

返回值

无

注意事项

无

示例

```
--形式一
Vec = LXVector3.new(1, 2, 3)
Vec1 = LXVector3.new(1, 1, 1)
Vec2 = LXVector3.new(1, 0, 1)
Vec3 = LXVector3.new(2, 5, 9)
Vec.CreateNormal(Vec1, Vec2, Vec3)
x,y,z = Vec.GetValue()
print(x)		--输出-0.992278
print(y)		--输出 0
print(z)		--输出 0.124035
--形式二
Vec = LXVector3.new(1, 2, 3)
Vec1 = LXVector3.new(1, 1, 1)
Vec2 = LXVector3.new(1, 0, 1)
Vec.CreateNormal(Vec1, Vec2)
x,y,z = Vec.GetValue()
print(x)		--输出 0.707
print(y)		--输出 0
print(z)		--输出-0.707
```

参见

Samples\Test_LuaBase\Test_LXVector3\脚本\XMain.lua

LXVector4

功能描述

该对象实现四维向量的数学运算及操作。

属性列表

属性名称	说明
x	LXVector4 的 x 属性
y	LXVector4 的 y 属性
z	LXVector4 的 z 属性
w	LXVector4 的 w 属性

函数列表

函数名称	说明
new	创建一个 LXVector4 对象
“+”运算符	两个向量相加
“-”运算符	两个向量相减
“*”运算符	缩放向量
“/”运算符	缩放向量
“==” 运算符	判断两个向量是否绝对相等
Absolute	分量取绝对值
Invert	分量取倒数
Negate	分量取负数
Normalize	规则化向量
Length	得到向量长度
Scale	缩放向量
Distance	计算两个向量之间的距离
DotProduct	计算两向量点乘
CrossProduct	计算两向量叉乘
SetLength	设置向量的长度
Lerp	线性插值
ComputeHeadingAndPitch	计算向量在 XZ 平面上的投影角度和 XY 平面上的投影角度
SetFromHeadingAndPitch	通过投影角度构造向量
SetFromSphericalCoords	通过球体经纬度构造向量
Reflect	计算基于法平面的反射向量
Refract	计算基于法平面的折射向量
Project	计算向量在法线上的投影
CreateNormal	根据给定的顺时针 3 点，或者给定的两向量，计算该平面法线
SetX	设置 X 分量
SetY	设置 Y 分量
SetZ	设置 Z 分量
SetW	设置 W 分量
GetX	获取 X 分量
GetY	获取 Y 分量
GetZ	获取 Z 分量
GetW	获取 W 分量
SetValue	设置 Vector4 的值
GetValue	获取 Vector4 的值

new

创建一个 LXVector4 对象

```
new(x, y, z, w)
new(Vec)
```

参数

参数名	类型	描述	是否可选	默认值
x	number	四维向量 x 值	是	0
y	number	四维向量 y 值	是	0
z	number	四维向量 z 值	是	0
w	number	四维向量 w 值	是	0

参数名	类型	描述	是否可选	默认值
Vec	LXVector4	四维向量	否	无

返回值

类型	描述
LXVector4	LXVector4 对象

注意事项

无

示例

```
Vec = LXVector4.new()      -- x = 0 y = 0 z = 0 w =5
Vec1 = LXVector4.new (-2,3,-4, 5)          --x = -2 y = 3 z = -4 w = 5
Vec2 = LXVector4.new(Vec1)
```

参见

Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua

“+”运算符

两个向量相加

```
Vector3 = Vector1 + Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector 1	LXVector4 对象	操作数	否	无
Vector 2	LXVector4 对象	操作数	否	无

返回值

无

注意事项

无

示例

```
Vector3 = Vector1 + Vector2
```

参见

Samples\Test_LuaBase\Test_LXVector4\脚本\XMain.lua

“-”运算符

两个向量相减

```
Vector3 = Vector1 - Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector 1	LXVector4 对象	操作数	否	无
Vector 2	LXVector4 对象	操作数	否	无

返回值

无

注意事项

无

示例

```
Vector3 = Vector1 - Vector2
```

参见

Samples\Test_LuaBase\Test_LXVector4\脚本\XMain.lua

“*”运算符

向量的乘运算

```
Vector3 = Vector1 *f
Vector3 = Vector1 * Vector2
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector4 对象	操作数	否	无
Vector2	LXVector4 对象	操作数	否	无
f	number	操作数	否	无

返回值

无

注意事项

若两个操作数都是向量时，调用 CrossProduct 函数

示例

```
Vector3 = Vector1 *0.1
Vector3 = Vector1 * Vector2
```

参见

Samples\Test_LuaBase\Test_LXVector4\脚本\XMain.lua

“/”运算符

缩放向量

```
Vector3 = Vector1 /f
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector4 对象	操作数	否	无
f	number	操作数	否	无

返回值

无

注意事项

无

示例

```
Vector3 = Vector1 /10
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\脚本\XMain.lua
```

“==”运算符

判断两个向量是否绝对相等

```
B = Vector3 == Vector1
```

参数

参数名	类型	描述	是否可选	默认值
Vector1	LXVector4 对象	操作数	否	无
Vector3	LXVector4 对象	操作数	否	无

返回值

类型	描述
boolean	true 表肯定， false 表否定

注意事项

```
无
```

示例

```
B = Vector3 == Vector1
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\脚本\XMain.lua
```

Absolute

分量取绝对值

```
Absolute()
```

参数

```
无
```

返回值

无

注意事项

无

示例

```
Vec.SetValue(-2,3,-4, 5)        --x = -2 y = 3 z = -4 w = 5
Vec.Absolute()
x,y,z,w = Vec.GetValue()
print(x)              --输出 2
print(y)              --输出 3
print(z)              --输出 4
print(w)               --输出 5
```

参见

Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua

Invert

分量取倒数

```
Invert()
```

参数

无

返回值

无

注意事项

无

示例

```
Vec = LXVector4.new()
Vec.SetValue(-2,3,-4, 5)        --x = -2 y = 3 z = -4 w = 5
Vec.Invert()
x,y,z,w = Vec.GetValue()
print(x)              --输出-0.5
print(y)              --输出 0.333333
```

```
print(z)                --输出-0.25
print(w)                 --输出 0.2
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

Negate

分量取负数

```
Negate()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
Vec = LXVector4.new()
Vec.SetValue(-2,3,-4, 5)          --x = -2 y = 3 z = -4 w = 5
Vec.Negate()
x,y,z,w = Vec.GetValue()
print(x)                --输出 2
print(y)                --输出-3
print(z)                --输出 4
print(w)                 --输出-5
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

Normalize

规则化向量

```
Normalize()
```

参数

无

返回值

无

注意事项

无

示例

```
Vec = LXVector4.new()
Vec.SetValue(-2,3,-4, 5)        --x = -2 y = 3 z = -4 w = 5
Vec.Normalize()
x,y,z,w = Vec.GetValue()
print(x)                --输出-0.2
print(y)                --输出 0.4
print(z)                --输出-0.5
print(w)                 --输出 0.6
```

参见

Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua

Length

得到向量长度

```
Length()
```

参数

无

返回值

类型	描述
number	向量长度

注意事项

无

示例

```
Vec = LXVector4.new()
Vec.SetValue(-2,3,-4, 5)        --x = -2 y = 3 z = -4 w = 5
res = Vec.Length()
print(res)              --输出 7.34
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

Scale

缩放向量

```
Scale(var)
```

参数

参数名	类型	描述	是否可选	默认值
var	number	缩放倍数	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vec = LXVector4.new()
Vec.SetValue(-2,3,-4, 5)        --x = -2 y = 3 z = -4 w = 5
Vec.Scale(2)
x,y,z,w = Vec.GetValue()
print(x)              --输出-4
print(y)              --输出 6
print(z)              --输出-8
print(w)              --输出 10
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

Distance

计算两向量距离

```
Distance(vec)
```

参数

参数名	类型	描述	是否可选	默认值
vec	LXVector4	目标向量	否	无

返回值

类型	描述
number	距离

注意事项

无

示例

```
Vec = LXVector4.new()      -- x = 0 y = 0 z = 0 w = 0
Vec1 = LXVector4.new (-2,3,-4,5)   --x = -2 y = 3 z = -4 w = 5
res = Vec.Distance(Vec1)
print(res)              --输出 7.34
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

DotProduct

计算两向量的点乘

```
DotProduct(vec)
```

参数

参数名	类型	描述	是否可选	默认值
vec	LXVector4	目标向量	否	无

返回值

类型	描述
number	点乘结果

注意事项

无

示例

```
Vec = LXVector4.new()      -- x = 0 y = 0 z = 0
Vec1 = LXVector4.new (-2,3,-4,5)   --x = -2 y = 3 z = -4 w=5
res = Vec.DotProduct(Vec1)
print(res)                --输出 0
```

参见

Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua

CrossProduct

计算两个或三个四维向量的叉乘

```
CrossProduct(vec1, vec2)
CrossProduct(vec1, vec2, vec3)
```

参数

参数名	类型	描述	是否可选	默认值
Vec1	LXVector4	向量 1	否	无
Vec2	LXVector4	向量 2	否	无

参数名	类型	描述	是否可选	默认值
Vec1	LXVector4	向量 1	否	无
Vec2	LXVector4	向量 2	否	无
Vec3	LXVector4	向量 3	否	无

返回值

无

注意事项

无

示例

```
--该函数计算两个或三个向量的叉乘，将结果保存在调用对象中
--形式一
Vec = LXVector4.new()
```

```
Vec1 = LXVector4.new(1, 2, 2, 2)
Vec2 = LXVector4.new(1, 2, 3, 4)
Vec.CrossProduct(Vec1, Vec2)
x,y,z , w= Vec.GetValue()
print(x)                --输出 0
print(y)                --输出 0
print(z)                --输出 0
print(w)               --输出 0
--形式二
Vec = LXVector4.new()
Vec1 = LXVector4.new(8, 4, 2, 0)
Vec2 = LXVector4.new(1, 2, 3, 0)
Vec3 = LXVector4.new(-4, 3, 5, 0)
Vec.CrossProduct(Vec1, Vec2, Vec3)
x,y,z , w= Vec.GetValue()
print(x)                --输出 0
print(y)                --输出 0
print(z)                --输出 0
print(w)               --输出 38
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

SetLength

设置向量长度

```
SetLength(fLength)
```

参数

参数名	类型	描述	是否可选	默认值
fLength	number	向量长度	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Vector.SetLength(10)            --设置向量长度为 10
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

Lerp

线性插值

```
Lerp(vec1, fVar)
Lerp(vec1, vec2, fVar)
```

参数

参数名	类型	描述	是否可选	默认值
Vec1	LXVector4	向量 1	否	无
fVar	number	插值参数	否	无

参数名	类型	描述	是否可选	默认值
Vec1	LXVector4	向量 1	否	无
Vec2	LXVector4	向量 2	否	无
fVar	number	插值参数	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
--形式一
Vec = LXVector4.new()
Vec1 = LXVector4.new(1, 2, 2, 2)
Vec.Lerp(Vec1, 0.9)
x,y,z , w= Vec.GetValue()
print(x)                --输出 0.9
print(y)                --输出 1.8
print(z)                --输出 1.8
print(w)               --输出 1.8
--形式二
Vec = LXVector4.new()
```

```
Vec1 = LXVector4.new(1, 2, 2, 2)
Vec2 = LXVector4.new(2, 3, 0, 1)
Vec.Lerp(Vec1, Vec2, 0.9)
x,y,z , w= Vec.GetValue()
print(x)                --输出 1.9
print(y)                --输出 2.9
print(z)                --输出 0.2
print(w)               --输出 1.1
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

ComputeHeadingAndPitch

计算向量在 XZ 和 XY 平面上的投影角度

```
ComputeHeadingAndPitch()
```

参数

```
无
```

返回值

类型	描述
number	XZ 平面投影角度
number	XY 平面投影角度

注意事项

```
无
```

示例

```
Vec = LXVector4.new(1, 1, 1, 2)
heading, pitch = Vec.ComputeHeadingAndPitch()
print(heading)                --输出-0.785
print(pitch)                  --输出 1.57
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

SetFromHeadingAndPitch

通过投影角度构造向量

```
SetFromHeadingAndPitch(Heading, Pitch)
```

参数

参数名	类型	描述	是否可选	默认值
Heading	number	XZ 平面投影角度	否	无
Pitch	number	XY 平面投影角度	否	无

返回值

无

注意事项

无

示例

```
Vec = LXVector4.new()
Vec.SetFromHeadingAndPitch(-0.78, 1.570)
x,y,z,w = Vec.GetValue()
print(x)              --输出-0.00056
print(y)              --输出-1
print(z)              --输出 0.000566
print(w)             --输出 0
```

参见

Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua

SetFromSphericalCoords

通过球体经纬度构造向量

```
SetFromSphericalCoords(longitude, latitude)
```

参数

参数名	类型	描述	是否可选	默认值
longitude	number	经度	否	无
latitude	number	纬度	否	无

返回值

无

注意事项

无

示例

```
Vec = LXVector4.new()
Vec.SetFromHeadingAndPitch (10, 20)
x,y,z,w = Vec.GetValue()
print(x)              --输出-0.222
print(y)              --输出-0.912
print(z)              --输出-0.342
print(w)               --输出 0
```

参见

Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua

Reflect

计算基于法平面的反射向量

```
Reflect(vecNormal)
```

参数

参数名	类型	描述	是否可选	默认值
vecNormal	LXVector3	参考法平面	否	无

返回值

无

注意事项

无

示例

```
Vec = LXVector4.new(1, 2, 3, 1)
VecNormal = LXVector3.new(1, 1, 1)
Vec.Reflect(VecNormal)
```

```
x,y,z,w = Vec.GetValue()
print(x)              --输出 11
print(y)              --输出 10
print(z)              --输出 9
print(w)              --输出 0
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

Refract

计算基于法平面的折射向量

```
Refract(vecNormal, fIncident, fTransmitte)
```

参数

参数名	类型	描述	是否可选	默认值
vecNormal	LXVector3	参考法平面	否	无
fIncident	number	入射介质系数	是	1
fTransmitte	number	折射介质系数	是	1

返回值

```
无
```

注意事项

```
无
```

示例

```
Vec = LXVector4.new(1, 2, 3,0)
VecNormal = LXVector3.new(1, 1, 1)
Vec.Refract(VecNormal, 1.0, 0.5)
x,y,z,w = Vec.GetValue()
print(x)              --输出 2.12
print(y)              --输出 4.12
print(z)              --输出 6.12
print(w)              --输出 0
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

Project

计算向量在法线上的投影

```
Project(vec)
```

参数

参数名	类型	描述	是否可选	默认值
vec	LXVector3	参考法线	否	无

返回值

无

注意事项

无

示例

```
Vec = LXVector4.new(1, 2, 3,0)
VecNormal = LXVector3.new(1, 1, 1)
Vec.Project(VecNormal)
x,y,z,w = Vec.GetValue()
print(x)                --输出 6
print(y)                --输出 6
print(z)                --输出 6
print(w)               --输出 0
```

参见

```
Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua
```

CreateNormal

根据顺时针三个点或者两个向量计算由它们确定的平面法线

```
CreateNormal(vec1, vec2, vec3)
```

参数

参数名	类型	描述	是否可选	默认值
vec1	LXVector4	参考点 1	否	无
vec2	LXVector4	参考点 2	否	无
vec3	LXVector4	参考点 3	否	无

参数名	类型	描述	是否可选	默认值
Vec1	LXVector4	指定向量 1	否	无
Vec2	LXVector4	指定向量 2	否	无

返回值

无

注意事项

无

示例

```
--形式一
Vec = LXVector4.new(1, 2, 3,4)
Vec1 = LXVector4.new(1, 1, 1,0)
Vec2 = LXVector4.new(1, 0, 1,0)
Vec3 = LXVector4.new(2, 5, 9,0)
Vec.CreateNormal(Vec1, Vec2, Vec3)
x,y,z,w = Vec.GetValue()
print(x)                --输出-0.992278
print(y)                --输出 0
print(z)                --输出 0.124035
print(w)               --输出 0
--形式二
Vec = LXVector4.new(1, 2, 3,0)
Vec1 = LXVector4.new(1, 1, 1,0)
Vec2 = LXVector4.new(1, 0, 1,0)
Vec.CreateNormal(Vec1, Vec2)
x,y,z ,w= Vec.GetValue()
print(x)                --输出 0.707
print(y)                --输出 0
print(z)                --输出-0.707
print(w)                 --输出 0
```

参见

Samples\Test_LuaBase\Test_LXVector4\Test_LXVector4.lua

3.6.2 场景对象

LXSceneCamera

功能描述

该对象实现对场景相机功能封装。

函数列表

函数名称	说明
SetViewMatrix	设置 3 维世界视觉坐标系
SetProjMatrix	设置透视矩阵
SetCameraPos	设置相机位置
GetCameraPos	得到相机位置
GetLookAt	得到相机绝对方向
TestPointVisible	测试指定点是否在当前相机的视锥内
TestSphereVisible	测试指定球体是否在当前相机的视锥内
TestBoxVertexVisible	测试指定立方体是否在当前相机的视锥内
TestBoxVisible	测试指定包围体是否在当前相机的视锥内
TestContainerVisible	测试指定包容器是否在当前相机的视锥内
GetRayFromScreenPt	转化屏幕坐标为 3 维空间坐标系射线
GetScreenPtFromVector	转化 3 维空间坐标顶点为屏幕坐标
GetScreenVectorFromVector	转化 3 维空间坐标顶点为 2 维空间顶点
SetAdjustAspectFlag	设置是否自动调整纵横比
GetAdjustAspectFlag	获取是否自动调整纵横比
RotateCamera	旋转相机
StartRotate	开始旋转相机
EndRotate	结束旋转相机
ChangeDistance	改变相机偏移位置
OffsetPos	改变相机位置
AttachToItem	将当前相机附加到指定 Item

SetViewMatrix

设置 3 维世界视觉坐标系

```
SetViewMatrix(Eye, LookAt, Up)
```

参数

参数名	类型	描述	是否可选	默认值
Eye	LXVector3	视点坐标	否	无
LookAt	LXVector3	视点方向	否	无
Up	LXVecotr3	上方向	否	无

返回值

类型	描述
Boolean	设置是否成功

注意事项

无

示例

```
vecEye = LXVector3.new(0,0,0)
vecLookAt = LXVector3.new(10,10,10)
vecUp = LXVector3.new(0,0,100)
bRes = camera.SetViewMatrix(vecEye, vecLookAt, vecUp)
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

SetProjMatrix

设置透视矩阵

```
SetProjMatrix(Fov, Aspect, NearPlane, FarPlane)
```

参数

参数名	类型	描述	是否可选	默认值
Fov	number	视宽	否	无
Aspect	number	纵横比	否	无
NearPlane	number	近面	否	无
FarPlane	number	远面	否	无

返回值

类型	描述

Boolean	设置是否成功

注意事项

无

示例

```
Fov = math.pi/4
Aspect= 1
NearPlane = 1
FarPlane = 10000
bRes = camera.SetProjMatrix(Fov, Aspect, NearPlane, FarPlane)
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

SetCameraPos

设置相机位置

```
SetCameraPos(Pos)
```

参数

参数名	类型	描述	是否可选	默认值
Pos	LXVector3	相机位置	否	无

返回值

无

注意事项

无

示例

```
Pos = LXVector3.new(0,0,0)
camera.SetCameraPos(Pos)
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

GetCameraPos

得到相机位置

```
GetCameraPos()
```

参数

无

返回值

类型	描述
LXVecotr3	相机位置

注意事项

无

示例

```
Pos = camera.GetCameraPos()
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

GetLookAt

得到相机绝对方向

```
GetLookAt()
```

参数

无

返回值

类型	描述
LXVecotr3	相机绝对方向

注意事项

无

示例

```
LookAt = camera.GetLookAt()
```

参见

```
Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua
```

TestPointVisible

测试指定点是否位于当前相机的视锥内

```
TestPointVisible(vecPoint)
```

参数

参数名	类型	描述	是否可选	默认值
vecPoint	LXVector3	指定的坐标点	否	无

返回值

类型	描述
Boolean	是否位于视锥内

注意事项

```
无
```

示例

```
vecPoint = LXVector3.new(0,0,0)
camera.TestPointVisible(vecPoint)
```

参见

```
Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua
```

TestSphereVisible

测试指定球体是否位于当前相机的视锥内

```
TestSphereVisible(Point, Radius)
```

参数

参数名	类型	描述	是否可选	默认值

vecPoint	LXVector3	球心坐标	否	无
Raius	number	半径	否	无

返回值

类型	描述
Boolean	是否位于视锥内

注意事项

无

示例

```
vecPoint = LXVector3.new(0,0,0)
Radius = 1000
camera.TestSphereVisible(vecPoint, Radius)
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

TestBoxVertexVisible

测试指定立方体是否位于当前相机的视锥内

```
TestBoxVertexVisible(vecPoint)
```

参数

参数名	类型	描述	是否可选	默认值
vecPoint	table	立方体顶点坐标集	否	无

返回值

类型	描述
Boolean	是否位于视锥内

注意事项

无

示例

```
vecPoint1 = LXVector3.new(0,0,0)
vecPoint2 = LXVector3.new(10,0,0)
vecPoint3 = LXVector3.new(10,0,10)
vecPoint4 = LXVector3.new(0,0,10)
vecPoint5 = LXVector3.new(0,10,0)
vecPoint6 = LXVector3.new(10,10,0)
vecPoint7 = LXVector3.new(10,10,10)
vecPoint8 = LXVector3.new(0,10,10)
vecPoint = {vecPoint1, vecPoint2, vecPoint3, vecPoint4, vecPoint5, vecPoint6, vecPoint7, vecPoint8}
bRes = camera.TestBoxVertexVisible(vecPoint)
print(bRes)
```

参见

```
Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua
```

TestBoxVisible

测试指定包围体是否位于当前相机的视锥内

```
TestBoxVisible(AABBBox)
TestBoxVisible(BoundBox)
```

参数

参数名	类型	描述	是否可选	默认值
AABBBox	LXAABBBox	包围体对象	否	无

参数名	类型	描述	是否可选	默认值
BoundBox	LXBoundBox	包围体对象	否	无

返回值

类型	描述
Boolean	是否位于视锥内

注意事项

```
无
```

示例

```
AABBBox = LXAABBBox.new()
```

```
AABBBox.SetAABBBox(0,0,0,100,100,100)
bRes = camera.TestAABBBoxVisible(AABBBox)
print(bRes)

BoundBox = LXBoundBox.new()
BoundBox.InitX3DBoundBox()
bRes = camera.TestBoundBoxVisible(BoundBox)
print(bRes)
```

参见

```
Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua
```

TestContainerVisible

测试指定包容器是否位于当前相机的视锥内

```
TestBoxVertexVisible(Point)
```

参数

参数名	类型	描述	是否可选	默认值
Point	table	包容器顶点坐标集	否	无

返回值

类型	描述
Boolean	是否位于视锥内

注意事项

```
无
```

示例

```
vecPoint1 = LXVector3.new(0,0,0)
vecPoint2 = LXVector3.new(10,0,0)
vecPoint3 = LXVector3.new(10,0,10)
vecPoint4 = LXVector3.new(0,0,10)
vecPoint5 = LXVector3.new(0,10,0)
vecPoint6 = LXVector3.new(10,10,0)
vecPoint7 = LXVector3.new(10,10,10)
vecPoint8 = LXVector3.new(0,10,10)
vecPoint = {vecPoint1, vecPoint2, vecPoint3, vecPoint4, vecPoint5, vecPoint6, vecP
```

```
oint7, vecPoint8}
bRes = camera.TestContainerVisible(vecPoint)
print(bRes)
```

参见

```
Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua
```

GetRayFromScreenPt

转化屏幕坐标为 3 维空间坐标系射线

```
GetRayFromScreenPt(Point)
```

参数

参数名	类型	描述	是否可选	默认值
Point	LXPoint	屏幕坐标点	否	无

返回值

类型	描述
LXVecotr3	射线原点
LXVecotr3	射线方向

注意事项

```
无
```

示例

```
Point = LXPoint.new(10,10)
vecOri, vecDir = camera.GetRayFromScreenPt(pt)
```

参见

```
Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua
```

GetScreenPtFromVector

转化 3 维空间坐标顶点为屏幕坐标

```
GetScreenPtFromVector(vecPoint)
```

参数

参数名	类型	描述	是否可选	默认值

vecPoint	LXVecotr3	3 维空间坐标	否	无

返回值

类型	描述
LXPoint	屏幕坐标

注意事项

无

示例

```
vecPoint = LXVector3.new(100,100,100)
Point = camera.GetScreenPtFromVector(vecPoint)
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

GetScreenVectorFromVector

转化 3 维空间坐标顶点为 2 维空间坐标顶点

```
GetScreenVectorFromVector(vecPoint)
```

参数

参数名	类型	描述	是否可选	默认值
vecPoint	LXVector3	3 维空间坐标顶点	否	无

返回值

类型	描述
LXVector2	2 维空间坐标顶点

注意事项

无

示例

```
vecPoint = LXVector3.new(100,100,100)
Point = camera.GetScreenVectorFromVector(vecPoint)
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

SetAdjustAspectFlag

设置是否自动调整纵横比

```
SetAdjustAspectFlag(bAuto)
```

参数

参数名	类型	描述	是否可选	默认值
bAuto	Boolean	是否自动调整	否	无

返回值

无

注意事项

无

示例

参见 GetAdjustAspectFlag

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

GetAdjustAspectFlag

获得是否自动调整纵横比

```
GetAdjustAspectFlag()
```

参数

无

返回值

类型	描述
Boolean	是否自动调整

注意事项

无

示例

```
bRes = camera.GetAdjustAspectFlag()
print(bRes)
if bRes == true then
	camera.SetAdjustAspectFlag(false)
else
	camera.SetAdjustAspectFlag(true)
end
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

RotateCamera

旋转相机

```
RotateCamera(pt)
```

参数

参数名	类型	描述	是否可选	默认值
pt	LXPoint	平面的点	否	无

返回值

类型	描述
Boolean	旋转成功返回 true，否则 false

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

StartRotate

开始旋转相机

StartRotate(pt)

参数

参数名	类型	描述	是否可选	默认值
pt	LXPoint	平面的点	否	无

返回值

类型	描述
Boolean	开始旋转成功返回 true，否则 false

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

EndRotate

结束相机旋转

EndRotate(pt)

参数

参数名	类型	描述	是否可选	默认值
pt	LXPoint	平面的点	否	无

返回值

类型	描述
Boolean	结束旋转成功返回 true，否则 false

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

ChangeDistance

改变相机偏移位置

```
ChangeDistance(dist)
```

参数

参数名	类型	描述	是否可选	默认值
dist	number	偏移距离	否	无

返回值

类型	描述
Boolean	偏移成功返回 true，否则 false

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

OffsetPos

改变相机位置

```
OffsetPos(vec3)
```

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	相机位置	否	无

返回值

类型	描述
boolean	成功改变相机位置返回 true，否则返回 false

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

AttachToItem

将当前相机附加到指定 Item

```
AttachToItem(Item)
```

参数

参数名	类型	描述	是否可选	默认值
Item	LXSceneItem	场景中指定的物体	否	无

返回值

无

注意事项

无

示例

```
Item = LXSceneItem.new()
Item.CreateX3DSimpleBoxItem(“Item”, 100, 100, 100)
camera.AttachToItem(Item)
```

参见

Samples\Test_3DScene\Test_LXSceneCamera\脚本\scene.lua

LXSceneManager

功能描述

该对象实现场景管理器。

函数列表

函数名称	说明
SetSceneManagerActive	设置场景管理器是否活动
GetSceneItemFromName	从名称获取场景物体
SetAmbientColor	设置场景材质环境光颜色
GetAmbientColor	获取场景材质光颜色
GetWaterHeight	得到地图块水平面高度
SetWaterHeight	设置地图块水平面高度
ReflectRenderIsEnable	得到当前场景是否允许渲染反射纹理
EnableReflectRender	设置当前场景是否允许渲染反射纹理
GetTerrain	获取当前场景地表块
GetSkybox	获取当前场景天空盒
GetTerPtFromClientPt	得到当前鼠标点下对应场景地表坐标
GetItemFrmPt	得到当前鼠标点下场景中碰撞的最近物体
LoadScript	装载指定的 lua 文件代码
UpdateSceneVisibleObjects	根据指定摄像机更新场景的可见物体列表
GetSceneCamera	获取当前场景相机
SetSceneCamera	设置当前场景相机
CreateSceneItem	创建场景物体
EnableRenderTerrain	设置是否渲染地表
EnableRenderSkybox	设置是否渲染天空体

SetSceneManagerActive

设置场景管理器是否活动

```
SetSceneManagerActive(bActive)
```

参数

参数名	类型	描述	是否可选	默认值
bActive	boolean	是否激活标志	否	无

返回值

无

注意事项

bActive
false 表示非激活
true 表示激活

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetSceneItemFromName

从名称获取场景物体

GetSceneItemFromName(name)

参数

参数名	类型	描述	是否可选	默认值
name	string	物体名称	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

SetAmbientColor

设置场景材质环境光颜色

SetAmbientColor(color)

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色	否	无

返回值

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetAmbientColor

获取场景材质光颜色

GetAmbientColor()

参数

无

返回值

类型	描述
LXColor	材质光颜色

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetWaterHeight

得到地图块水平面高度

GetWaterHeight()

参数

无

返回值

类型	描述
number	地表块水平面高度

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

SetWaterHeight

设置地图块水平面高度

SetWaterHeight(height)

参数

参数名	类型	描述	是否可选	默认值
height	number	地表块水平面高度	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

ReflectRenderIsEnable

得到当前场景是否允许渲染反射纹理

ReflectRenderIsEnable()

参数

无

返回值

类型	描述
boolean	是否允许渲染反射纹理

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

EnableReflectRender

设置当前场景是否允许渲染反射纹理

EnableReflectRender(bEnable)

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否允许	否	无

返回值

注意事项

bEnable
false　表示不允许
true　表示允许

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetTerrain

获取当前场景地表块

GetTerrain()

参数

无

返回值

类型	描述
LXSceneTerrain	地表块

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetSkybox

获取当前场景天空盒

GetSkybox()

参数

无

返回值

类型	描述
LXSceneSkyBox	天空盒

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetTerPtFromClientPt

得到当前鼠标点下对应场景地表坐标

```
GetTerPtFromClientPt(point)
```

参数

参数名	类型	描述	是否可选	默认值
point	LXPoint	鼠标位置	否	无

返回值

类型	描述
LXPoint	场景地表坐标

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetItemFrmPt

得到当前鼠标点下场景中碰撞的最近物体

```
GetItemFrmPt(point, fMinLimit)
```

参数

参数名	类型	描述	是否可选	默认值

point	LXPoint	鼠标位置	否	无
fMinLimit	number	最小限制距离	否	无

返回值

类型	描述
LXSceneItem	场景物体

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

LoadScript

装载指定的 lua 文件代码

```
LoadScript(name)
```

参数

参数名	类型	描述	是否可选	默认值
name	string	Lua 文件名	否	无

返回值

类型	描述
boolean	是否成功

注意事项

false 表示失败
true 表示成功

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

UpdateSceneVisibleObjects

根据指定摄像机更新场景的可见物体列表

UpdateSceneVisibleObjects(scenecamera)

参数

参数名	类型	描述	是否可选	默认值
scenecamera	LXSceneCamera	场景相机	否	无

返回值

类型	描述
number	是否成功

注意事项

0 表示成功
非 0 表示失败

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

GetSceneCamera

获取当前场景相机

GetSceneCamera()

参数

无

返回值

类型	描述
LXSceneCamera	场景相机

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

SetSceneCamera

设置当前场景相机

```
SetSceneCamera(camera)
```

参数

参数名	类型	描述	是否可选	默认值
camera	LXSceneCamera	场景相机	否	无

返回值

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

CreateSceneItem

创建场景物体

```
CreateSceneItem(name, modelname, bonename)
```

参数

参数名	类型	描述	是否可选	默认值
name	string	物体名称	否	无
modelname	string	物体模型文件	否	无
bonename	string	物体骨架文件	是	无

返回值

类型	描述
LXSceneItem	场景物体

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

EnableRenderTerrain

是否允许渲染地表

```
EnableRenderTerrain(bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否允许渲染地表	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

EnableRenderSkybox

是否允许渲染天空体

```
EnableRenderSkybox (bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否允许渲染天空体	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneManager\脚本\scene.lua

LXSceneItem

功能描述

该对象实现场景物体功能封装。

函数列表

函数名称	说明
AttacheSceneItemModel	给指定场景对象附加一个模型
AttachSceneItemBoneHierarchy	给指定场景对象附加一个骨架
CreateSceneItem	创建一个基本物品
CreateX3DSimpleBoxItem	创建一个立方体场景对象
CreateX3DPrecisionBoxItem	创建一个指定精度的立方体场景对象
CreateX3DSphereItem	创建一个球体场景对象
CreateX3DHalfSphereItem	创建一个半球体场景对象
CreateX3DCylinderItem	创建一个圆柱体或圆锥或圆台场景对象
CreateX3DPlaneItem	创建一个 3 维平面场景对象
CreateX3DPolygonItem	创建一个指定半径的等边多边形平面场景对
CreateX3DSegmentItem	创建一个 3 维线段数组场景对象
CreateX3DCircleItem	创建一个 3 维圆圈场景对象

CreateX3DTorusItem	创建一个 3 维圆环场景对象
Create3DAxisItem	创建一个 3 维坐标轴场景对象
Create3DTranslateAxisItem	创建一个 3 维坐标轴平移场景对象
Create3DScaleAxisItem	创建一个 3 维坐标轴缩放场景对象
Create3DRotateAxisItem	创建一个 3 维坐标轴旋转场景对象
GetComponentCount	得到物体部件总数
GetScale	得到物体缩放因子
SetScale	设置物体缩放因子
GetAttachNode	得到物体挂接的节点
SetAttachNode	设置物体挂接的节点
GetRenderQueneLayer	得到物体渲染层次
SetRenderQueneLayer	设置物体渲染层次
GetID	得到物体 ID
SetID	设置物体 ID
GetVisible	得到物体是否可见
SetVisible	设置物体是否可见
GetBoundBoxRenderType	得到物体包围盒显示方式
SetBoundBoxRenderType	设置物体包围盒显示方式
GetBoundColor	得到物体包围网格的颜色
SetBoundColor	设置物体包围网格的颜色
GetPosition	得到物体的位置
SetPosition	设置物体的位置
GetLookAt	得到物体的方向
MoveInDir	沿着当前方向平移
LiftUpDir	沿着当前上方向平移
LeftRightDir	沿着当前方向向左平移
SetDirection	设置节点方向
SetRotate	设置物体旋转
GetRotate	获得物体当前的方位
EnableForwardCamera	设置物体是否朝向相机
IsForwardCamera	获得物体是否朝向相机
EnableVerticakCamera	设置物体朝向相机时，是否与相机垂直
IsVerticakCamera	获得物体朝向相机时，是否与相机垂直
Rotate	旋转物体
IsNode	获得对象是否是节点
IsItem	获得对象是否是物体
IsVisible	获得对象针对指定相机是否可见
TestInterSectDirection	测试当前物体是否与指定射线相交
TestInterSectAABBBox	测试当前物体是否与指定包围盒发生碰撞
GetObjectType	获得对象类型
ClearAllLight	清除物体应用灯光信息
GetLightCount	得到当前的灯光总数

GetLightDistance	得到指定索引的灯光距离
DestroyObject	删除物体
SetCheckBoneValidate	设置是否检测骨架的有效性
SetItemAnimateSetSpeed	设置当前播放的动画集的播放速度
SetLoopPlayBoneAnimateFlag	设置是否循环播放骨骼动画
GetLoopPlayBoneAnimateFlag	获得是否循环播放骨骼动画
SetItemAnimateSet	设置当前播放的动画
GetPlayAnimateEndedFlag	获得当前动画是否播放完毕
SetPlayAnimateEndedFlag	设置当前动画是否播放完毕
GetPlayAnimateTimePoint	获取当前动画的播放时刻
SetPlayAnimateTimePoint	设置当前动画的播放时刻
GetAnimatePlayType	获取当前动画的播放方式
SetAnimatePlayType	设置当前动画的播放方式
ResetItemAnimateSet	重置当前动画

AttachSceneItemModel

给指定场景对象附加一个模型

```
AttachSceneItemModel(FileName)
```

参数

参数名	类型	描述	是否可选	默认值
FileName	string	模型文件名	否	无

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

如果指定场景对象已经拥有一个模型，则替换该模型

示例

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

AttachSceneItemBoneHierarchy

给指定场景对象附加一个骨架

```
AttachSceneItemBoneHierarchy(FileName)
```

参数

参数名	类型	描述	是否可选	默认值
FileName	string	骨架文件名	否	无

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

如果指定场景对象已经拥有一个骨架，则替换该骨架

示例

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateSceneItem

创建一个基本物品

```
CreateSceneItem(SceneManager, Name, ModelName, BoneName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
ModelName	string	指定的模型文件	是	nil
BoneName	string	指定的骨架文件	是	nil

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

无

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.CreateSceneItem(this, “Item”)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DSimpleBoxItem

创建一个立方体场景对象

```
CreateX3DSimpleBoxItem(SceneManager, Name, Width, Height, Depth, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
Width	number	立方体宽	否	无
Height	number	立方体高	否	无
Depth	number	立方体深度	否	无
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VertexType 取值如下:
= 1　　顶点 + Normal
= 2　　顶点 + Diffuse + Specular
= 3　　顶点 + Normal + uv
= 4　　顶点 + Diffuse + Specular + uv
= 5　　顶点 + Normal + Diffuse + Specular + uv
= 6　　顶点 + Normal + uv + 混合 1 个骨架
= 7　　顶点 + Normal + uv + 混合 2 个骨架
= 8　　顶点 + Normal + uv + 混合 3 个骨架
= 9　　顶点 + Normal + uv + 混合 4 个骨架
= 10　顶点 + Diffuse + Specular + AddColor1 + AddColor2 + uv
= 11　经过处理的顶点
= 12　经过处理的顶点 + Diffuse + Specular
= 13　经过处理的顶点 + uv
= 14　经过处理的顶点 + Diffuse + Specular + uv
= 15　经过处理的顶点 + uv2
= 16　经过处理的顶点 + Diffuse + Specular + uv2
= 17　经过处理的顶点 + Diffuse + uv
= 18　经过处理的顶点 + Diffuse + uv2
= 19　顶点 + Diffuse + uv

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.CreateX3DSimpleBoxItem(this, "Item", 1024, 1024, 1024)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DPrecisionBoxItem

创建一个指定精度的立方体场景对象

CreateX3DSimpleBoxItem(SceneManager, Name, Width, Height, Depth, Precision, VertexType, InnerName)

参数

参数名	类型	描述	是否可选	默认值

SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
Width	number	立方体宽	否	无
Height	number	立方体高	否	无
Depth	number	立方体深度	否	无
Precision	number	精度	是	8
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item. CreateX3DSimpleBoxItem(this, "Item", 1024, 1024, 1024)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DSphereItem

创建一个球体场景对象

```
CreateX3DSphereItem(SceneManager, Name, Radius, Precision, MinU, MaxU, MinV, MaxV, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无

Radius	number	球体半径	否	无
MinU	number	最小 U 坐标	是	0
MaxU	number	最大 U 坐标	是	1
MinV	number	最小 V 坐标	是	0
MaxV	number	最大 V 坐标	是	0.5
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item. CreateX3DSphereItem(this, "Item", 1024)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DHalfSphereItem

创建一个半球体场景对象

```
CreateX3DHalfSphereItem(SceneManager, Name, Radius, Precision, MinU, MaxU, MinV, MaxV, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
Radius	number	球体半径	否	无
Precision	number	精度	是	8

MinU	number	最小 U 坐标	是	0
MaxU	number	最大 U 坐标	是	1
MinV	number	最小 V 坐标	是	0
MaxV	number	最大 V 坐标	是	0.5
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

无

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item. CreateX3DHalfSphereItem(this, "Item", 1024)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DCylinderItem

创建一个圆柱体或圆锥或圆台场景对象

```
CreateX3DCylinderItem(SceneManager, Name, BaseRadius, UpRadius, Height, HPrecision, VPrecision, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
BaseRadius	number	底半径	否	无
UpRadius	number	顶半径	否	无
Height	number	高度	否	无

HPrecision	number	水平精度	是	8
VPrecision	number	垂直精度	是	8
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item. CreateX3DCylinderItem(this, "Item", 1024, 768, 600)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DPlaneItem

创建一个 3 维平面场景对象

```
CreateX3DPlaneItem(SceneManager, Name, Width, Height, Precision, bVertical, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
Width	number	宽度	否	无
Height	number	高度	否	无
Precision	number	平面精度	是	8
bVertical	Boolean	是否与水平面垂直	是	false

VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.CreateX3DPlaneItem(this, "Item", 1024, 1024)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DPolygonItem

创建一个指定半径的等边多边形平面场景对象

CreateX3DPolygonItem(SceneManager, Name, Radius, Precision, DirPrecision, bVertical, VertexType, InnerName)

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
Radius	number	平面半径	否	无
Precision	number	平面精度	否	无
DirPrecision	number	径向精度	否	无
bVertical	Boolean	是否与水平面垂直	是	false
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name

				参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.CreateX3DPolygonItem(this, "Item", 1024, 8, 8, true)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DSegmentItem

创建 3 维线段数组场景对象

```
CreateX3DSegmentItem(SceneManager, Name, Count, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
Count	number	线段数量	否	无
VecPoint	table	线段顶点数组	否	无
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

线段顶点数组成员个数应等于线段数量的 2 倍

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
vecPoint1 = LXVector3.new(0,0,0)
vecPoint2 = LXVector3.new(10,0,0)
vecPoint3 = LXVector3.new(10,0,10)
vecPoint4 = LXVector3.new(0,0,10)
vecPoint5 = LXVector3.new(0,10,0)
vecPoint6 = LXVector3.new(10,10,0)
vecPoint7 = LXVector3.new(10,10,10)
vecPoint8 = LXVector3.new(0,10,10)
vecPoint = {vecPoint1, vecPoint2, vecPoint3, vecPoint4, vecPoint5, vecPoint6, vecPoint7, vecPoint8}
Item.CreateX3DSegmentItem(this, "Item", 4, vecPoint)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DCircleItem

创建 3 维圆圈场景对象

```
CreateX3DCircleItem(SceneManager, Name, Count, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
Radius	number	圆圈半径	否	无
Precision	number	圆圈精度	否	无
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

无

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.CreateX3DCircleItem(this, "Item", 100, 8)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

CreateX3DTorusItem

创建 3 维圆环场景对象

```
CreateX3DTorusItem(SceneManager, Name, InnerRadius, OuterRadius, Precision, SidePrecision, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
InnerRadius	number	圆环的内径	否	无
OuterRadius	number	圆环的外径	否	无
Precision	number	圆环精度	否	无
SidePrecision	number	圆环轴精度	否	无
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.CreateX3DTorusItem(this, "Item", 20, 100, 8, 8)
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

Create3DAxisItem

创建一个 3 维坐标轴场景对象

```
Create3DAxisItem(SceneManager, Name, ConeRadius, CylinderRadius, ConeHeight, CylinderHeigth, HPrecision, VPrecision, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
ConeRadius	number	箭头圆锥半径	是	16
CylinderRadius	number	箭头圆柱半径	是	6
ConeHeight	number	箭头圆锥高度	是	50
CylinderHeight	number	箭头圆柱高度	是	300
HPrecision	number	圆面精度	是	8
VPrecision	number	垂直精度	是	4
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

成功返回 0，失败返回错误码

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.Create3DAxisItem(this, "Item")
end
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

Create3DTranslateAxisItem

创建一个 3 维坐标轴平移场景对象

Create3DTranslateAxisItem(SceneManager, Name, ConeRadius, CylinderRadius, ConeHeight, CylinderHeigth, HPrecision, VPrecision, VertexType, InnerName)

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
ConeRadius	number	箭头圆锥半径	是	16
CylinderRadius	number	箭头圆柱半径	是	6
ConeHeight	number	箭头圆锥高度	是	50
CylinderHeight	number	箭头圆柱高度	是	300
HPrecision	number	圆面精度	是	8
VPrecision	number	垂直精度	是	4
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

VectexType 参数说明见 CreateX3DSimpleBoxItem

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.Create3DTranslateAxisItem(this, "Item")
end
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

Create3DScaleAxisItem

创建一个 3 维坐标轴缩放场景对象

```
Create3DScaleAxisItem(SceneManager, Name, ConeRadius, CylinderRadius, ConeHeight, CylinderHeigth, HPrecision, VPrecision, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
ConeRadius	number	箭头圆锥半径	是	16
CylinderRadius	number	箭头圆柱半径	是	6
ConeHeight	number	箭头圆锥高度	是	50
CylinderHeight	number	箭头圆柱高度	是	300
HPrecision	number	圆面精度	是	8
VPrecision	number	垂直精度	是	4
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

```
VectexType 参数说明见 CreateX3DSimpleBoxItem
```

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
Item = LXSceneItem.new()
Item.Create3DScaleAxisItem(this, "Item")
end
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

Create3DRotateAxisItem

创建一个 3 维坐标轴旋转场景对象

```
Create3DRotateAxisItem(SceneManager, Name, InnerRadius, OuterRadius, Precision, SidePrecision, VertexType, InnerName)
```

参数

参数名	类型	描述	是否可选	默认值
SceneManeger	LXSceneManager	场景管理器对象	否	无
Name	string	物品名称	否	无
InnerRadius	number	圆环的内径	否	无
OuterRadius	number	圆环的外径	否	无
Precision	number	圆环精度	否	无
SidePrecision	number	圆环轴精度	否	无
VertexType	number	顶点缓冲类型	是	3
InnerName	string	内部名称	是	与 Name 参数相同

返回值

类型	描述
number	绘制成功返回 0，失败返回非 0

注意事项

```
VectexType 参数说明见 CreateX3DSimpleBoxItem
```

示例

```
--this 为系统传入的场景管理器对象
function  OnLoadLua(this,nEventID,SendName,param)
```

```
Item = LXSceneItem.new()
Item.Create3DRotateAxisItem(this, "Item")
end
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

GetComponentCount

得到物体部件总数

```
GetComponentCount()
```

参数

```
无
```

返回值

类型	描述
number	部件总数

注意事项

```
无
```

示例

```
Count = Item.GetComponentCount()
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

GetScale

得到物体缩放因子

```
GetScale()
```

参数

```
无
```

返回值

类型	描述

LXVector3	缩放因子

注意事项

无

示例

```
vec = Item.GetScale()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetScale

设置物体缩放因子

```
SetScale(vec)
```

参数

参数名	类型	描述	是否可选	默认值
vec	LXVector3	缩放因子	否	无

返回值

无

注意事项

无

示例

```
vec = LXVector3.new()
Item.SetScale(vec)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetAttachNode

得到物体挂接的节点

GetAttachNode()

参数

无

返回值

类型	描述
LXSceneNode	物体挂接的节点对象

注意事项

无

示例

Node = Item.GetAttachNode()

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetAttachNode

设置物体挂接的节点

SetAttachNode()

参数

参数名	类型	描述	是否可选	默认值
Node	LXSceneNode	节点对象	否	无

返回值

无

注意事项

无

示例

Item.SetAttachNode(Node)

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetRenderQueneLayer

得到物体渲染层次

GetRenderQueneLayer()

参数

无

返回值

类型	描述
number	渲染层次

注意事项

参数说明见 SetRenderQueneLayer 函数

示例

Layer = Item.GetRenderQueneLayer()

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetRenderQueneLayer

设置物体渲染层次

SetRenderQueneLayer(Layer)

参数

参数名	类型	描述	是否可选	默认值
Layer	number	渲染层次	否	无

返回值

无

注意事项

Layer 取值如下:

```
= 0        未定义层次
= 1        远景/背景渲染层
= 2        地表渲染层
= 3        地表装饰层
= 4        物体渲染层
= 5        物体附加层
= 6        地表水渲染层
= 7        透明对象渲染层
= 8        该层开始清空先前的深度
= 9        窗口对象渲染层
= 10       顶层渲染层
```

示例

```
Layer = 1
Item.SetRenderQueneLayer(Layer)
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

GetID

得到物体 ID

```
GetID()
```

参数

```
无
```

返回值

类型	描述
number	物体 ID

注意事项

```
无
```

示例

```
ID = Item.GetID()
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

SetID

设置物体 ID

```
SetID(ID)
```

参数

参数名	类型	描述	是否可选	默认值
ID	number	物体 ID	否	无

返回值

无

注意事项

无

示例

```
ID = 10001
Item.SetID(ID)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetVisible

得到物体是否可见

```
GetVisible()
```

参数

无

返回值

类型	描述
Boolean	物体是否可见

注意事项

无

示例

```
bVisible = Item.GetVisible()
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

SetVisible

设置物体是否可见

```
SetVisible(bVisible)
```

参数

参数名	类型	描述	是否可选	默认值
bVisible	Boolean	物体是否可见	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
bVisible = true
Item.SetVisible(bVisible)
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

GetBoundBoxRenderType

得到物体包围盒显示方式

```
GetBoundBoxRenderType()
```

参数

```
无
```

返回值

类型	描述

number	物体包围盒显示方式

注意事项

参数说明见 SetBoundBoxRenderType 函数

示例

```
Type = Item.GetBoundBoxRenderType()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetBoundBoxRenderType

设置物体包围盒显示方式

```
SetBoundBoxRenderType(Type)
```

参数

参数名	类型	描述	是否可选	默认值
Type	string	显示方式	是	nil

返回值

无

注意事项

Type 可以是下面几种类型的组合:
“a” 显示轴对齐的包围盒
“b” 显示非轴对齐的包围盒
“d” 显示包围网格
“m” 显示骨架网格
“r” 显示为网格，如果没有则显示实体
“n” 不显示模型网格
默认值为 nil，即不显示包围盒

示例

```
Type = “abdm”
Item.SetBoundBoxRenderType(Type)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetBoundColor

得到物体包围网格的颜色

```
GetBoundColor()
```

参数

无

返回值

类型	描述
LXColor	物体包围网格的颜色

注意事项

无

示例

```
Color = Item.GetBoundColor()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetBoundColor

设置物体包围网格的颜色

```
SetBoundColor(Color)
```

参数

参数名	类型	描述	是否可选	默认值
Color	LXColor	包围网格的颜色	否	无

返回值

类型	描述
Boolean	设置是否成功

注意事项

无

示例

```
Color = LXColor.new(255,255,0)
bRes = Item.SetBoundColor(Color)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetPosition

得到物体的位置

```
GetPosition()
```

参数

无

返回值

类型	描述
LXVector3	物体位置

注意事项

无

示例

```
Vec = Item.GetPosition()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetPosition

设置物体的位置

```
SetPosition(Vec)
```

参数

参数名	类型	描述	是否可选	默认值
vec	LXVector3	物体位置	否	无

返回值

无

注意事项

无

示例

```
Vec = LXVector3.new()
Item.SetPosition(Vec)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetLookAt

得到物体的方向

```
GetLookAt()
```

参数

无

返回值

类型	描述
LXVector3	物体方向

注意事项

无

示例

```
Vec = Item.GetLookAt()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

MoveInDir

沿着当前方向平移

```
MoveInDir(Length)
```

参数

参数名	类型	描述	是否可选	默认值
Length	number	平移距离	否	无

返回值

无

注意事项

无

示例

```
Length = 100
Item.MoveInDir(Length)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

LiftUpDir

沿着当前上方向平移

```
LiftUpDir(Length)
```

参数

参数名	类型	描述	是否可选	默认值
Length	number	平移距离	否	无

返回值

无

注意事项

无

示例

```
Length = 100
Item.LiftUpDir(Length)
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

LeftRightDir

沿着当前方向向左平移

```
LeftRightDir(Length)
```

参数

参数名	类型	描述	是否可选	默认值
Length	number	平移距离	否	无

返回值

```
无
```

注意事项

```
无
```

示例

```
Length = 100
Item.LeftRightDir(Length)
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

SetDirection

设置节点方向

```
SetDirection(Vec)
```

参数

参数名	类型	描述	是否可选	默认值
Vec	LXVector3	节点方向	否	无

返回值

```
无
```

注意事项

无

示例

```
Vec = LXVector3.new(100,100,100)
Item.SetDirection(Vec)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetRotate

设置物体旋转

```
SetRotate(Quaternion)
```

参数

参数名	类型	描述	是否可选	默认值
Quaternion	LXQuaternion	旋转因子	否	无

返回值

无

注意事项

无

示例

```
Quaternion = LXQuaternion.new(100,100,100,100)
Item.SetRotate(Quaternion)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetRotate

获得物体当前的方位

```
GetRotate()
```

参数

无

返回值

类型	描述
LXQuaternion	旋转因子

注意事项

无

示例

```
Quaternion = Item.GetRotate()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

EnableForwardCamera

设置物体是否朝向相机

```
EnableForwardCamera(bEnable)
```

参数

参数名	类型	描述	是否可选	默认值
bEnable	Boolean	是否朝向相机	否	无

返回值

无

注意事项

无

示例

```
bEnable = true
Item.EnableForwardCamera(bEnable)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

IsForwardCamera

获得物体是否朝向相机

```
IsForwardCamera()
```

参数

无

返回值

类型	描述
Boolean	物体是否朝向相机

注意事项

无

示例

```
bEnable = Item.IsForwardCamera()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

EnableVerticakCamera

设置物体朝向相机时，是否与相机垂直

```
EnableVerticakCamera(bVert)
```

参数

参数名	类型	描述	是否可选	默认值
bVert	Boolean	是否朝向相机	否	无

返回值

无

注意事项

无

示例

```
bVert = true
Item.EnableForwardCamera(bVert)
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

IsVerticakCamera

获得物体朝向相机时，是否与相机垂直

```
IsVerticakCamera()
```

参数

无

返回值

类型	描述
Boolean	物体是否垂直相机

注意事项

无

示例

```
bVert = Item.EnableForwardCamera(bVert)
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

Rotate

旋转物体

```
Rotate(Axis, Angle)
```

参数

参数名	类型	描述	是否可选	默认值
Axis	number	旋转类型	否	无
Angle	number	旋转角度	否	无

返回值

无

注意事项

Axis 取值如下:
= 1　　绕 X 轴旋转
= 2　　绕 Y 轴旋转
= 3　　绕 Z 轴旋转

示例

```
Item.Rotate(1， 90)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

IsNode

获得对象是否是节点

```
IsNode()
```

参数

无

返回值

类型	描述
Boolean	对象是否是节点

注意事项

无

示例

```
bNode = Item.IsNode()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

IsItem

获得对象是否是物体

```
IsItem()
```

参数

无

返回值

类型	描述
Boolean	对象是否是物体

注意事项

无

示例

```
bItem = Item.IsItem()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

IsVisible

获得对象针对指定相机是否可见

```
IsVisible(camera)
```

参数

camera	LXSceneCamera	相机对象

返回值

类型	描述
Boolean	对象针对指定相机是否可见

注意事项

无

示例

```
bVisible = Item.IsVisible(camera)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

TestInterSectDirection

测试当前物体是否与指定射线相交

```
TestInterSectDirection(Ori, Dir, Limit, CollideMask)
```

参数

参数名	类型	描述	是否可选	默认值
Ori	LXVector3	射线原点位置	否	无
Dir	LXVector3	射线方向	否	无
Limit	number	距离限制	否	无
CollideMask	string	碰撞检测控制	否	无

返回值

类型	描述
table	返回结果集合

注意事项

CollideMask 以使下面几种类型的组合:
“a” 返回所有碰撞的物体，不能与”o”一起使用
“c” 检测物体自身网格 否则采用碰撞网格进行检测
“e” 检测地表时使用当前显示级别检测，否则使用实际数据
“i” 检测物体碰撞 否则不检测物体碰撞
“o” 仅返回碰撞最近的物体 否则随机返回一个碰撞物体 不能与”a”一起使用
“p” 计算物体碰撞最近点 否则仅返回物体上碰撞点不保证最近
“t” 检测地表碰撞 否则不检测地表碰撞
“w” 检测水体碰撞 否则不检测水体碰撞

返回值 Res 内容如下:
table[“Distance”] 交点距离，大于 0 表示相交
table[“CollidePoint”] 碰撞点坐标
table[“CollideNormal”] 碰撞位置法线
table[“ComponentIndex”] 碰撞部件索引
table[“FaceIndex”] 碰撞面索引
table[“Node”] 碰撞节点对象

示例

```
function OnLButtonDown(this, nEventID, SendName, param)
local Point = LXPoint.new(param.x, param.y)
local vecOri, vecDir = camera.GetRayFromScreenPt(Point)
Res = Item.TestInterSectDirection(vecOri, vecDir, 1000, “ace”)
end
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

TestInterSectAABBBox

测试当前物体是否与指定包围盒发生碰撞

```
TestInterSectAABBBox(Box)
```

参数

参数名	类型	描述	是否可选	默认值
Box	LXAABBBox	包围盒对象	否	无

返回值

类型	描述
Boolean	是否碰撞

注意事项

无

示例

```
vcMin = LXVector3.new(0,0,0)
vcMax = LXVector3.new(100,100,100)
Box = LXAABBBox.new()
Box.SetAABBBox(vcMin, vcMax)
bRes = Item.TestInterSectAABBBox(Box)
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

GetObjectType

获得对象类型

```
GetObjectType()
```

参数

```
无
```

返回值

类型	描述
number	对象类型

注意事项

```
Type 取值如下:
= 0		未定义
= 1		场景物体精灵
= 2		场景动态物体精灵
= 3		场景人物
= 4		场景灯光对象
```

示例

```
Type = Item.GetObjectType()
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

ClearAllLight

清除物体应用灯光信息

```
ClearAllLight()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
Item.ClearAllLight()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetLightCount

得到当前的灯光总数

```
GetLightCount()
```

参数

无

返回值

类型	描述
number	灯光总数

注意事项

无

示例

```
Count = Item.GetLightCount()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetLightDistance

得到指定索引的灯光距离

```
GetLightDistance(Index)
```

参数

参数名	类型	描述	是否可选	默认值
Index	number	灯光索引	否	无

返回值

类型	描述
number	指定索引灯光的距离

注意事项

无

示例

```
Distance = Item.GetLightDistance(0)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

DestroyObject

删除物体

```
DestroyObject()
```

参数

无

返回值

无

注意事项

无

示例

```
Item.DestroyObject()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetCheckBoneValidate

设置是否检测骨架的有效性

```
SetCheckBoneValidate(bValidate)
```

参数

参数名	类型	描述	是否可选	默认值
bValidate	Boolean	是否检测	否	无

返回值

无

注意事项

无

示例

```
Item.SetCheckBoneValidate(true)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetItemAnimateSetSpeed

设置当前播放的动画集的播放速度

```
SetItemAnimateSetSpeed(Speed)
```

参数

参数名	类型	描述	是否可选	默认值
Speed	number	播放速度	是	1

返回值

无

注意事项

无

示例

```
Item.SetItemAnimateSetSpeed()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetLoopPlayBoneAnimateFlag

设置是否循环播放骨骼动画

```
SetLoopPlayBoneAnimateFlag(bLoop)
```

参数

参数名	类型	描述	是否可选	默认值
bLoop	Boolean	是否循环播放	否	无

返回值

无

注意事项

无

示例

```
Item.SetLoopPlayBoneAnimateFlag(true)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetLoopPlayBoneAnimateFlag

获得是否循环播放骨骼动画

```
GetLoopPlayBoneAnimateFlag()
```

参数

无

返回值

类型	描述
Boolean	是否循环播放

注意事项

无

示例

```
bLoop = Item.GetLoopPlayBoneAnimateFlag()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetItemAnimateSet

设置当前播放的动画

```
SetItemAnimateSet(Handle)
```

参数

参数名	类型	描述	是否可选	默认值
Handle	LXHandle	动画句柄	否	无
bLoop	Boolean	是否循环播放	是	true

返回值

类型	描述
Boolean	是否播放成功

注意事项

无

示例

```
Handle = LXHandle.new()
Handle.Load("D:\\黄山松.xan")
if Handle.IsValidate() then
Item.SetItemAnimateSet(Handle)
end
Handle.CloseHandle()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetPlayAnimateEndedFlag

获得当前动画是否播放完毕

```
GetPlayAnimateEndedFlag()
```

参数

无

返回值

类型	描述

Boolean	动画是否播放完毕

注意事项

无

示例

```
bEnd = Item.GetPlayAnimateEndedFlag()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetPlayAnimateEndedFlag

设置当前动画是否播放完毕

```
SetPlayAnimateEndedFlag()
```

参数

参数名	类型	描述	是否可选	默认值
bEnd	Boolean	是否播放完毕	是	true

返回值

无

注意事项

无

示例

```
Item.SetPlayAnimateEndedFlag()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetPlayAnimateTimePoint

设置当前动画的播放时刻

```
SetPlayAnimateTimePoint(Time)
```

参数

参数名	类型	描述	是否可选	默认值
Time	number	指定的播放时刻	否	无

返回值

无

注意事项

播放时刻取值范围: 0 – 最大时刻

示例

```
Item.SetPlayAnimateTimePoint(50)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetPlayAnimateTimePoint

获得当前动画的播放时刻

```
GetPlayAnimateTimePoint()
```

参数

无

返回值

类型	描述
number	当前动画的播放时刻

注意事项

无

示例

```
Time = Item.GetPlayAnimateTimePoint()
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

SetAnimatePlayType

设置当前动画的播放方式

```
SetAnimatePlayType(Type)
```

参数

参数名	类型	描述	是否可选	默认值
Type	Boolean	播放方式	是	false

返回值

无

注意事项

Type = true 按时刻播放
Type = false 按帧顺序播放

示例

```
Item.SetAnimatePlayType(true)
```

参见

Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua

GetAnimatePlayType

获得当前动画的播放方式

```
GetAnimatePlayType()
```

参数

无

返回值

类型	描述
Boolean	当前动画的播放方式

注意事项

无

示例

```
Type = Item.GetAnimatePlayType()
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

ResetItemAnimateSet

重置当前动画

```
ResetItemAnimateSet()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```
Item.ResetItemAnimateSet()
```

参见

```
Samples\Test_3DScene\Test_LXSceneItem\脚本\scene.lua
```

LXSceneNode

功能描述

该对象实现场景节点，从 LXSceneItem 获取，无需自己创建

函数列表

函数名称	说明
Destroy	销毁场景节点
ClearAllChild	清除节点的所有孩子
RemoveChild	移除节点的孩子
AddChild	增加孩子节点
GetChild	获取孩子节点
GetChildCount	获取孩子节点个数
GetParent	获取节点父亲节点
SetParent	设置节点父亲节点
GetDerivedRotate	得到指定节点最终方位
GetDerivedPosition	得到指定节点最终位置
GetDerivedScale	得到指定节点最终缩放因子
GetMatrix	得到节点的变换矩阵
GetMatrixInverse	得到节点的逆变换矩阵
SetMatrix	强制设置节点的最终变换矩阵,忽略所有变换
Scale	缩放当前节点
SetScale	设置当前节点的缩放因子
GetScale	得到当前节点的缩放因子
RotateByQuaternion	根据指定的四元数旋转节点
SetRotate	设置节点旋转的四元数
GetRotate	得到节点旋转的四元数
Translation	转换节点位置
SetPosition	设置节点的位置
GetPosition	得到节点的位置
SetInheritOrientation	设置是否继承父节点的旋转
GetInheritOrientation	得到是否继承父节点的旋转标志
SetInheritScale	设置是否继承父节点的缩放
GetInheritScale	得到是否继承父节点的缩放
SetInitPos	设置初始节点位置(相对于父节点)
SetInitScale	设置初始节点缩放
SetInitRotate	设置初始节点旋转
DisableApplyInitNode	删除应用初始节点变换
TestInterSectDirection	指定节点是否和指定射线相交
GetAllDescendantCount	得到当前节点下所有子孙节点总数
MoveInDir	沿着当前方向平移指定距离
LiftUpDir	沿着当前上方向平移指定距离
LeftRightDir	沿着当前方向向左平移指定距离
Rotate	绕任意轴旋转节点
Pitch	绕 X 轴旋转节点
Yaw	绕 Y 轴旋转节点
Roll	绕 Z 轴旋转节点

SetDirection	设置节点的方向,上方向将根据当前的上方向
LookAt	设置 LookAt 方向
IsNode	是否为节点对象
IsItem	是否为物体对象
GetNodeColor	得到绘制节点颜色
SetNodeColor	设置绘制节点颜色
SetBelongItem	设置节点所属物体的指针,标明为物体的骨架节点
GetBelongItem	得到节点所属物体的指针
GetAttachItem	得到节点附加的物体的指针
SetAttachItem	设置节点附加的物体的指针
EnableForwardCamera	设置节点是否朝向相机
EnableVerticakCamera	设置节点朝向相机时,是否保持与相机垂直
GetEnableForwardCamera	得到节点是否朝向相机
GetEnableVerticakCamera	得到节点朝向相机时,是否保持与相机垂直
Detach	把该节点从当前场景脱离
Attach	把该节点重新挂接到指定场景

Destroy

创建场景物体

```
Destroy()
```

参数

无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

ClearAllChild

创建场景物体

```
ClearAllChild()
```

参数

无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

RemoveChild

创建场景物体

```
RemoveChild(name)
RemoveChild(index)
```

参数

参数名	类型	描述	是否可选	默认值
Name	string	节点名称	否	无
Index	number	索引	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

AddChild

增加孩子节点

AddChild(node)

参数

参数名	类型	描述	是否可选	默认值
Node	LXSceneNode	孩子节点	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetChild

获取孩子节点

GetChild(name, bSearchChildNode, bOnlySameItem)
GetChild(index, bSearchChildNode, bOnlySameItem)

参数

参数名	类型	描述	是否可选	默认值
Name	string	节点名称	否	无
Index	number	索引	否	无
bSearchChildNode	boolean	搜索子孩子	是	false
bOnlySameItem	boolean	是否相同的 item	是	false

返回值

类型	描述
LXSceneNode	场景节点

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetChildCount

获取孩子节点个数

GetChildCount()

参数

无

返回值

类型	描述
number	孩子节点个数

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetParent

获取父亲节点

GetParent()

参数

无

返回值

类型	描述
LXSceneNode	父亲节点

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetParent

设置父亲节点

SetParent(parentnode)

参数

参数名	类型	描述	是否可选	默认值
parentnode	LXSceneNode	父亲节点	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetDerivedRotate

得到指定节点最终方位

```
GetDerivedRotate()
```

参数

无

返回值

类型	描述
LXQuaternion	四元式

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetDerivedPosition

得到指定节点最终位置

```
GetDerivedPosition()
```

参数

无

返回值

类型	描述
LXVector3	节点位置

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetDerivedScale

得到指定节点最终缩放因子

GetDerivedScale()

参数

无

返回值

类型	描述
LXVector3	缩放因子

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetMatrix

得到节点的变换矩阵

GetMatrix()

参数

返回值

类型	描述
LXMatrix	变换矩阵

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetMatrixInverse

得到节点的逆变换矩阵

GetMatrixInverse()

参数

返回值

类型	描述
LXMatrix	逆变换矩阵

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetMatrix

设置节点的最终变换矩阵

SetMatrix(matrix)

参数

参数名	类型	描述	是否可选	默认值
matrix	LXMatrix	变换矩阵	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

Scale

缩放当前节点

Scale(vec3)

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	缩放因子	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetScale

设置当前节点的缩放因子

SetScale(vec3)

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	缩放因子	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetScale

获取当前节点的缩放因子

```
GetScale(mode)
```

参数

参数名	类型	描述	是否可选	默认值
mode	number	旋转方式	是	2

返回值

类型	描述
LXVector3	缩放因子

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

RotateByQuaternion

根据指定的四元数旋转节点

RotateByQuaternion(quaternion, mode)

参数

参数名	类型	描述	是否可选	默认值
quaternion	LXQuaternion	四元式	否	无
mode	number	旋转方式	是	1

返回值

无

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetRotate

设置节点旋转的四元数

SetRotate(quaternion, mode)

参数

参数名	类型	描述	是否可选	默认值
quaternion	LXQuaternion	四元数	否	无
mode	number	旋转方式	是	2

返回值

无

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetRotate

获取节点旋转的四元数

```
GetRotate(mode)
```

参数

参数名	类型	描述	是否可选	默认值
mode	number	旋转方式	是	2

返回值

类型	描述
LXQuaternion	四元数

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

```
Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua
```

Translation

转换节点位置

```
Translation(vec3, mode)
```

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	转换因子	否	无
mode	number	旋转方式	是	1

返回值

```
无
```

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

```

```

参见

```
Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua
```

SetPosition

设置节点的位置

```
SetPosition(vec3, mode)
```

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	节点位置	否	无
mode	number	旋转方式	是	2

返回值

无

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetPosition

得到节点的位置

```
GetPosition(mode)
```

参数

参数名	类型	描述	是否可选	默认值
mode	number	旋转方式	是	2

返回值

类型	描述
LXVector3	节点位置

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetInheritOrientation

设置是否继承父节点的旋转

SetInheritOrientation(bInherit)

参数

参数名	类型	描述	是否可选	默认值
bInherit	boolean	是否继承	否	无

返回值

无

注意事项

bInherit
false 表示不继承
true 表示继承

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetInheritScale

设置是否继承父节点的缩放

SetInheritScale(bInherit)

参数

参数名	类型	描述	是否可选	默认值
bInherit	boolean	是否继承	否	无

返回值

无

注意事项

bInherit
false 表示不继承
true 表示继承

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetInheritOrientation

得到是否继承父节点的旋转标志

```
GetInheritOrientation()
```

参数

无

返回值

类型	描述
number	是否继承

注意事项

返回值
1 表示继承
0 表示不继承

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetInheritScale

得到是否继承父节点的缩放

```
GetInheritScale()
```

参数

无

返回值

类型	描述
number	是否继承父亲的缩放

注意事项

返回值
1 表示继承
0 表示不继承

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetInitPos

设置初始节点位置

SetInitPos(vec3)

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	节点位置	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetInitScale

设置初始节点缩放

```
SetInitScale(vec3)
```

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	节点位置	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetInitRotate

设置初始节点旋转

```
SetInitRotate(quarternion)
```

参数

参数名	类型	描述	是否可选	默认值
quarternion	LXQuaternion	四元数	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

ResetInit

重置当前节点为初始状态

```
ResetInit(bReset)
```

参数

参数名	类型	描述	是否可选	默认值
bReset	boolean	是否重置	否	无

返回值

无

注意事项

bReset
true 表示重置
false 表示不重置

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetInitPos

得到当前初始节点位置

```
GetInitPos()
```

参数

无

返回值

类型	描述

LXVector3	节点初始化位置

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetInitScale

得到当前初始节点缩放

```
GetInitScale()
```

参数

无

返回值

类型	描述
LXVector3	节点初始化缩放

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetInitRotate

得到当前初始节点旋转

```
GetInitRotate()
```

参数

无

返回值

类型	描述
LXQuaternion	四元数

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

DisableApplyInitNode

删除应用初始节点变换

```
DisableApplyInitNode()
```

参数

无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

TestInterSectDirection

指定节点是否和指定射线相交

```
TestInterSectDirection(vec3Org, vec3Dir, fMinLimit, collidemask)
```

参数

参数名	类型	描述	是否可选	默认值
vec3Org	LXVector3	射线原点	否	无
vec3Dir	LXVector3	射线方向	否	无
fMinLimit	number	最小限制距离	否	无
collidemask	string	相交标志	否	无

返回值

类型	描述
LXSceneItem	场景物体

注意事项

```
Collidemask  可以以下串一起组合
"P"        //计算物体碰撞最近点否则仅返回物体上碰撞点,不保证最近,但速度较快
"N"        //仅返回碰撞最近的物体,否则随机返回一个碰撞物体,该项不能和"A"一块使用
"A"        //返回所有碰撞的物体,该项不能和"N"一块使用
"M"        //检测物体自身网格否则采用碰撞网格进行检测
"I"        //检测物体碰撞否则不检测物体碰撞
"T"        //检测地表碰撞否则不检测地表碰撞
"W"        //检测水体碰撞否则不检测水体碰撞
"L"        //检测地表时使用当前显示级别检测,否则使用实际数据监测
```

示例

参见

```
Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua
```

GetAllDescendantCount

得到当前节点下所有子孙节点总数

```
GetAllDescedantCount()
```

参数

无

返回值

类型	描述
number	子孙节点个数

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

MoveInDir

沿着当前方向平移指定距离

```
MoveInDir(fDist)
```

参数

参数名	类型	描述	是否可选	默认值
fDist	number	平移距离	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

LiftUpDir

沿着当前上方向平移指定距离

```
LiftUpDir(fDist)
```

参数

参数名	类型	描述	是否可选	默认值
fDist	number	平移距离	否	无

返回值

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

LeftRightDir

沿着当前方向向左平移指定距离

```
LeftRightDir(fDist)
```

参数

参数名	类型	描述	是否可选	默认值
fDist	number	平移距离	否	无

返回值

无

注意事项

无

示例

参见

```
Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua
```

Rotate

绕任意轴旋转节点

```
Rotate(vec3, fAngle, mode)
Rotate(quaternion, mode)
```

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	旋转向量	否	无
fAngle	number	旋转角度	否	无
mode	number	旋转模式	是	1
quaternion	LXQuaternion	旋转四元数	否	无

返回值

```
无
```

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

```

```

参见

```
Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua
```

Pitch

绕 X 轴旋转节点

```
Pitch(fAngle, mode)
```

参数

参数名	类型	描述	是否可选	默认值

fAngle	number	旋转角度	否	无
mode	number	旋转模式	是	1

返回值

无

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

Yaw

绕 Y 轴旋转节点

```
Yaw(fAngle, mode)
```

参数

参数名	类型	描述	是否可选	默认值
fAngle	number	旋转角度	否	无
mode	number	旋转模式	是	1

返回值

无

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

Roll

绕 Z 轴旋转节点

Roll(fAngle, mode)

参数

参数名	类型	描述	是否可选	默认值
fAngle	number	旋转角度	否	无
mode	number	旋转模式	是	1

返回值

无

注意事项

mode
1 //变换相对于节点空间
2 //变换相对于父节点空间
3 //变换相对于世界坐标空间

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetDirection

设置节点的方向

SetDirection(vec3, bHorizon, mode)

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	节点方向	否	无

bHorizon	Boolean	是否水平	否	无
mode	number	变换方式	是	1

返回值

无

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

LookAt

设置 LookAt 方向

```
LookAt(vec3, bHorizon, mode)
```

参数

参数名	类型	描述	是否可选	默认值
vec3	LXVector3	节点方向	否	无
bHorizon	boolean	是否水平	否	无
mode	number	变换方式	是	1

返回值

无

注意事项

```
mode
1                   //变换相对于节点空间
2                   //变换相对于父节点空间
3                   //变换相对于世界坐标空间
```

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

IsNode

是否为节点对象

IsNode()

参数

无

返回值

类型	描述
boolean	是否为节点, true 表示是节点, false 表示不是节点

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

IsItem

是否为物体对象

IsItem()

参数

无

返回值

类型	描述
number	是否为物体, true 表示是物体, false 表示不是物体

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetNodeColor

得到绘制节点颜色

GetNodeColor()

参数

无

返回值

类型	描述
LXColor	节点颜色

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetNodeColor

设置绘制节点颜色

SetNodeColor(color)

参数

参数名	类型	描述	是否可选	默认值
color	LXColor	颜色	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetBelongItem

设置节点所属物体的指针

```
SetBelongItem(item)
```

参数

参数名	类型	描述	是否可选	默认值
item	LXSceneItem	物体	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetBelongItem

得到节点所属物体的指针

```
GetBelongItem()
```

参数

无

返回值

类型	描述
LXSceneItem	依附的物体

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetAttachItem

得到节点附加的物体的指针

```
GetAttachItem()
```

参数

无

返回值

类型	描述
LXSceneItem	附加物体

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

SetAttachItem

设置节点附加的物体的指针

SetAttachItem(item)

参数

参数名	类型	描述	是否可选	默认值
item	LXScenItem	附加物体	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

EnableForwardCamera

设置节点是否朝向相机

EnableForwardCamera(bEnable)

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否朝向相机	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

EnableVerticakCamera

设置节点朝向相机时,是否保持与相机垂直

EnableVerticakCamera(bEnable)

参数

参数名	类型	描述	是否可选	默认值
bEnable	boolean	是否朝向相机	否	无

返回值

类型	描述
LXSceneItem	场景物体

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetEnableForwardCamera

得到节点是否朝向相机

GetEnableForwardCamera()

参数

无

返回值

类型	描述
number	是否朝向相机

注意事项

返回值
true 表示朝向
false 表示不朝向

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

GetEnableVerticakCamera

得到节点朝向相机时,是否保持与相机垂直

```
GetEnableVerticakCamera()
```

参数

无

返回值

类型	描述
boolean	得到节点朝向相机时,是否保持与相机垂直

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

Detach

把该节点从当前场景脱离

```
Detach(bDetachFromParent)
```

参数

参数名	类型	描述	是否可选	默认值
bDetachFromParent	boolean	是否从父节点上脱离	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

Attach

把该节点重新挂接到指定场景

```
Attach(scenemanager, node)
```

参数

参数名	类型	描述	是否可选	默认值
scenemanager	LXSceneManager	场景管理器	否	无
node	LXSceneNode	节点	否	无

返回值

无

注意事项

无

示例

参见

Samples\Test_3DScene\Test_LXSceneNode\脚本\scene.lua

LXParticle

功能描述

该对象实现对场景粒子对象实例功能封装。

函数列表

函数名称	说明
new	创建一个粒子系统实例对象
UpdateParticle	更新粒子系统实例
DestroyParticle	销毁粒子对象实例
GetMatrixFromScreenPoint	根据屏幕上的点获取粒子变换矩阵

new

创建一个 LXPoint 对象

```
new(hParticle)
new(strParticleFileName)
```

参数

参数名	类型	描述	是否可选	默认值
hParticle	LXHandle	LXHandle 对象	否	无

参数名	类型	描述	是否可选	默认值
strParticleFileName	string	粒子系统实例文件路径	否	无

返回值

类型	描述
LXParticle	LXParticle 对象

注意事项

无

示例

```
hParticle = LXHandle.new("..\\资源\\snow.xpt","PARTICLE")
Particle1 = LXParticle.new(hParticle)

Particle2 = LXParticle.new("..\\资源\\yh.xpt")
```

参见

Samples\Test_3DScene\ \Test_LXParticle \脚本\scene.lua

UpdateParticle

更新粒子系统实例

```
UpdateParticle (sceneCamera, Martrix, Elapsed)
```

参数

参数名	类型	描述	是否可选	默认值
sceneCamera	LXSceneCamera	场景相机	否	无
Martrix	LXMatrix	变换矩阵	否	无
Elapsed	Number	更新时间	否	无

返回值

无

注意事项

sceneCamera = nil 表示应用默认的场景管理器的相机
该函数应用于场景更新函数中见示例

示例

```
hParticle = LXHandle.new("..\\资源\\snow.xpt","PARTICLE")
particle = LXParticle.new(hParticle)
Matrix = LXMatrix.new()
Matrix.Translation(5000,0,5000)
function   OnUpdateItem(this,nEventID,SendName,param)
```

```
    fElapsedtime = param;
    if particle ~= nil then
        particle.UpdateParticle(nil,Matrix,fElapsedtime)
    end
end
```

参见

```
Samples\Test_3DScene\ \Test_LXParticle \脚本\scene.lua
```

DestroyParticle

销毁粒子系统实例

```
DestroyParticle ()
```

参数

```
无
```

返回值

```
无
```

注意事项

```
无
```

示例

```

```

参见

```
Samples\Test_3DScene\ \Test_LXParticle \脚本\scene.lua
```

GetMatrixFromScreenPoint

根据屏幕上的点获取粒子系统实例的变换矩阵

```
GetMatrixFromScreenPoint (ScreenPoint, Distance)
```

参数

参数名	类型	描述	是否可选	默认值
ScreenPoint	LXPoint	屏幕上的点	否	无
Distance	Number	相机距离	是	1000

返回值

类型	描述
LXMatrix	粒子系统实例变换矩阵

注意事项

该函数主要应用于二维情况或者在当前视区中显示粒子实例的三维情况。 用户可以通过该函数获取屏幕上指定点放置粒子实例的变换矩阵

示例

```
hParticle = LXHandle.new("..\\资源\\yh.xpt","PARTICLE")
particle = LXParticle.new(hParticle)
point = LXPoint.new(500,700)
Matrix = particle.GetMatrixFromScreenPoint(point,2000)
function  OnUpdateItem(this,nEventID,SendName,param)
   fElapsedtime = param;
   if particle ~= nil then
       particle.UpdateParticle(nil,Matrix,fElapsedtime)
    end
end
```

参见

Samples\Test_3DScene\ \Test_LXParticle \脚本\scene.lua

3.7 事件

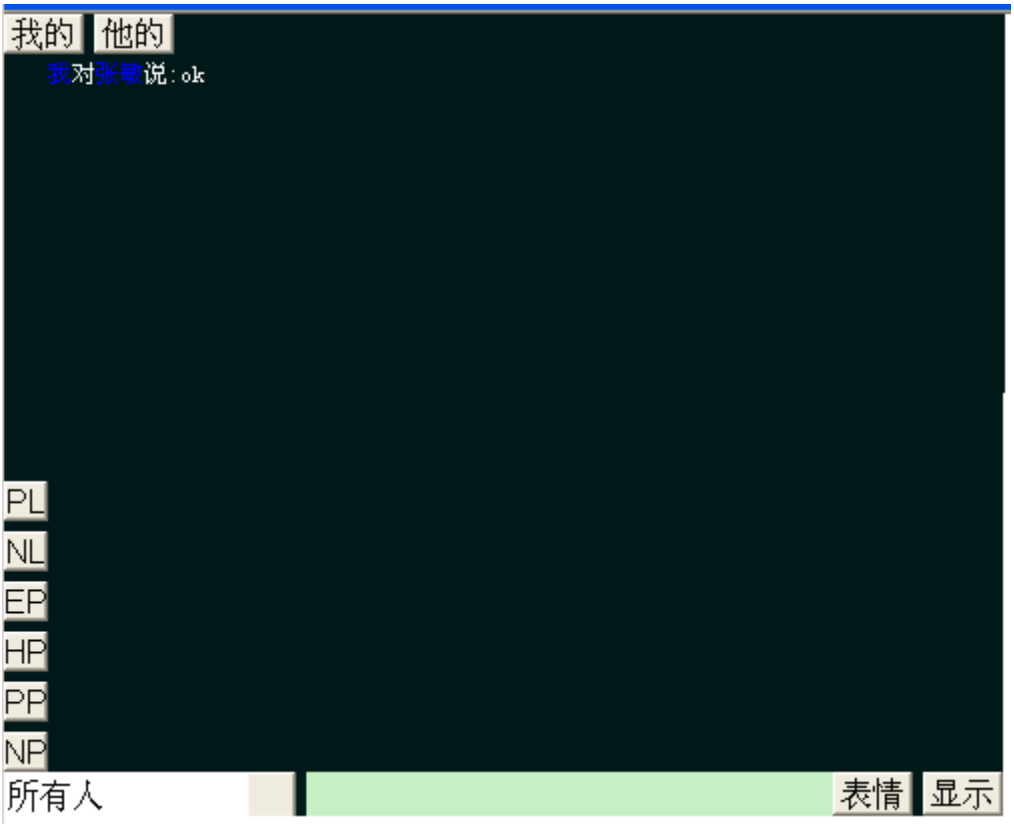
详细内容参考 Lua 事件。

3.8 高级应用模块

LXChatWnd

功能描述

该对象实现聊天功能的基本需求。该模块有 4 个部分组成.界面如下:



最上面的一排 我的 他的 为聊天内容过滤条件，可以对不同的人说的话进行过滤显示.。比如按选择我的按钮， 就显示所有关于我的聊天内容， 而不显示别人的聊天内容。底下一排 所有人 表情 显示 为聊天输入框， 所有人 这个可以控制聊天的对象，中间是聊天内容输入部分，可以输入你想说的话。右边的按钮用户可以根据需要进行添加和删除， 比如：表情按钮可以插入选择聊天需要的表情。显示按钮可以控制聊天窗口的显示与隐藏。左边一排为控制聊天内容显示（包含首页，末页，前一页，后一页，前一行，后一行），这些按钮用户可以根据需要进行添加和删除。 中间区域为聊天内容显示区。

使用事项

> 聊天窗口的消息发送给 NotifyWnd 窗口接收, NotifyWnd 窗口负责处理聊天窗口发送给它的消息，聊天窗口自己不处理这些消息.目的是为了让用户自己更好的控制这些消息.目前聊天窗口发送的消息有聊天输入框回车消息，按钮单击消息.

函数列表

函数名称	说明
CreateChatWnd	创建聊天窗口

AddUser	添加聊天者
RemoveUser	删除聊天者
RemoveAllUser	删除所有聊天者
SetCurUser	设置当前聊天者
GetCurUser	得到当前聊天者
SendContent	发送聊天内容
InsertStr	插入串
GetEditText	获取 edit 内容
HomePage	首页
NextPage	下一页
PrevPage	前一页
EndPage	尾页
NextLine	下一行
PrevLine	前一行
SetFilter	设置过滤条件
GetPanel	得到面板
AddPanelCtrl	添加 Panel 的子控件
RemovePanelCtrl	移除 Panel 的子控件
SetNotifyWnd	设置消息通知窗口

CreateChatWnd

创建聊天窗口

```
CreateChatWnd (rect, hParent)
```

参数

参数名	类型	描述	是否可选	默认值
rect	LXRect	聊天窗口位置	否	无
hParent	LXWnd	聊天窗口的父亲	是	O

返回值

无

事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

AddUser

添加聊天者

AddUser (strName, nIndex)

参数

参数名	类型	描述	是否可选	默认值
strName	string	表示聊天者的名字	否	无
nIndex	number	表示添加插入时的序号，-1 表示添加到末尾	是	-1

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

RemoveUser

删除聊天者

RemoveUser (strName)

参数

参数名	类型	描述	是否可选	默认值
strName	string	表示聊天者的名字	否	无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

RemoveAllUser

删除所有聊天者

```
RemoveAllUser ()
```

参数

无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

SetCurUser

设置当前聊天者

```
SetCurUser (strName)
```

参数

参数名	类型	描述	是否可选	默认值
strName	string	表示聊天者的名字	否	无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

GetCurUser

得到当前聊天者

```
GetCurUser ()
```

参数

无

返回值

类型	描述
string	表示聊天者的名字

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

SendContent

发送聊天内容

SendContent (dwLineSrcID， dwLineDestID， dwLineChatType， lpszLinePrefix，lpszText)

参数

参数名	类型	描述	是否可选	默认值
dwLineSrcID	number	表示发送者的 ID	否	无
dwLineDestID	number	表示接受者的 ID	否	无
dwLineChatType	number	表示聊天类型	否	无
lpszLinePrefix	string	表示聊天前缀	否	无
lpszText	string	表示聊天内容	否	无

返回值

无

注意事项

lpszLinePrefix 的格式为 xxx 对 yyy 说:
xxx 为发送者的名字
yyy 为接收者的名字

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

InsertStr

往聊天输入框中插入串

```
InsertStr (str, nPos)
```

参数

参数名	类型	描述	是否可选	默认值
str	string	插入的内容	否	无
nPos	number	插入位置,在 nPos 处插入串 ,nPos < 0 表示在当前位置插入 1	否	无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

GetEditText

获取聊天框内容

```
GetEditText ()
```

参数

无

返回值

类型	描述
string	表示文本内容

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

HomePage

翻到首页聊天内容

```
HomePage ()
```

参数

无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

NextPage

翻到下一页聊天内容

```
NextPage ()
```

参数

无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

PrevPage

翻到前一页聊天内容

PrevPage ()

参数

无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

EndPage

翻到最后一页聊天内容

```
EndPage ()
```

参数

无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

NextLine

翻到下一行聊天内容

```
NextLine ()
```

参数

无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

PrevLine

翻到前一行聊天内容

PrevLine ()

参数

无

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

SetFilter

设置聊天内容显示时的过滤条件

SetFilter (dwLineSrcID, dwLineDestID, dwLineChatType)

参数

参数名	类型	描述	是否可选	默认值
dwLineSrcID	number	发话者的 ID， 为 0 表示所有发话者	否	无
dwLineDestID	number	接受者的 ID，为 0 表示所有接受者	否	无
dwLineChatType	number	要显示的记录的聊	否	无

		天类型，为 0 表示所有聊天类型		

返回值

无

注意事项

无

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

GetPanel

得到指定位置的面板

```
GetPanel (nPanel)
```

参数

参数名	类型	描述	是否可选	默认值
nPanel	number	面板的序号	否	无

返回值

类型	描述
LXWnd	面板

注意事项

nPanel
0 表示聊天输入框面板
1 表示聊天内容滚动面板
2 表示聊天内容过滤面板

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

AddPanelCtrl

添加面板的子控件

```
AddPanelCtrl(nPanel, nID, lpszCaption, strStyle, rect)
```

参数

参数名	类型	描述	是否可选	默认值
nPanel	number	面板的序号	否	无
nID	number	面板控件的 ID	否	无
lpszCaption	string	面板控件的标题	否	无
strStyle	string	面板控件的风格，具体值参加 LXWnd 文档	否	无
rect	LXRect	表示面板控件的位置	否	无

返回值

类型	描述
LXWnd	表示控件对象

注意事项

nPanel
0 表示聊天输入框面板
1 表示聊天内容滚动面板
2 表示聊天内容过滤面板

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

RemovePanelCtrl

移除面板的子控件

```
RemovePanelCtrl (nPanel, nID)
```

参数

参数名	类型	描述	是否可选	默认值
nPanel	number	面板的序号	否	无
nID	number	面板控件的 ID	否	无

返回值

无

注意事项

nPanel
0 表示聊天输入框面板
1 表示聊天内容滚动面板
2 表示聊天内容过滤面板

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

SetNotifyWnd

设置消息通知窗口

```
SetNotifyWnd (LXWnd)
```

参数

参数名	类型	描述	是否可选	默认值
LXWnd	LXWnd	聊天窗口发送消息后的消息接受窗口	否	无

返回值

无

注意事项

聊天窗口的消息发送给 LXWnd, 这个窗口负责处理聊天窗口发送给它的消息. 目前有聊天输入框回车消息, 按钮单击消息

示例

无

参见

Samples\Test_LXChatWnd\Test_LXChatWnd\脚本\XMain.lua

4 应用示例

示例一

Hello World
此示例演示如何在输出中打印”Hello World”字符串

示例二

TestWnd
此示例演示窗口的创建，销毁，交互等

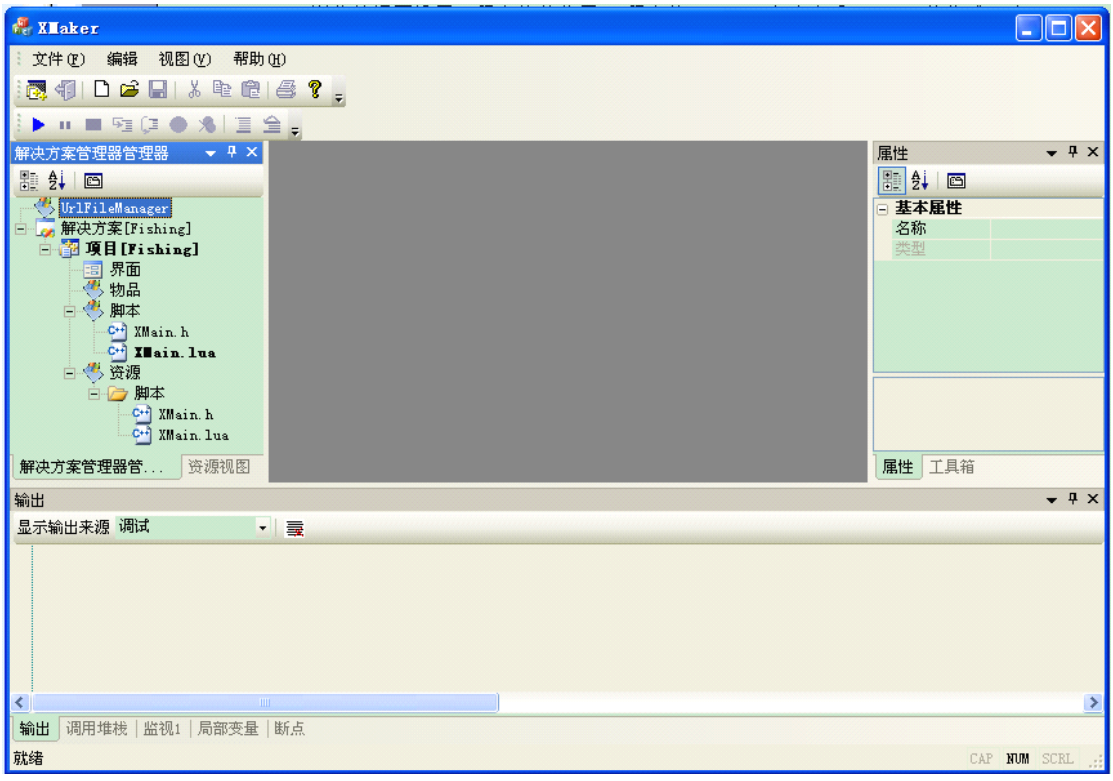
示例三

大鱼吃小鱼
此示例为一款简单休闲类 2D 游戏，演示 2D 应用
游戏截图：
游戏开始界面如下图：

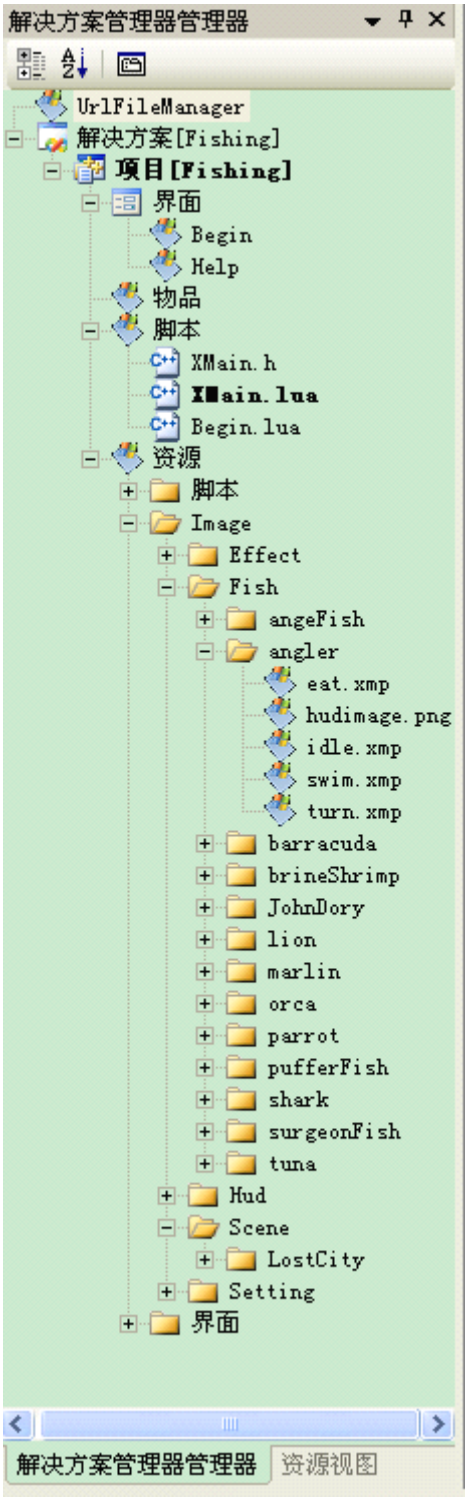
游戏运行图如下：

游戏工程项目的创建:

运行 XMaker 程序后,菜单上文件->新建->项目将弹出新建项目对话框（参阅工程项目创建),浏览按钮可设置工程存放的位置,工程名为 Fishing. 点确定后,XMaker 将生成一个 Fishing.xsln 项目如下图



游戏项目资源的添加:
在资源目录下添加各个目录并将资源添加到目录下如下图:



游戏开始界面:
界面包含新建游戏，游戏选项，高分纪录，退出游戏
该功能由 LXGame 类来实现,LXGame 为游戏开始界面类,主要包含 LXGame:HitTestPt 检测鼠标点和 LXGame:Draw 绘制界面函数
游戏运行:

游戏开始后关卡为 1,分数为零,玩家有三条生命值,吃比自己小的鱼将会增加自己的分数，分数到达一定鱼将会长大，从而能吃够吃到更高级的鱼。
游戏运行类包括 LXFishGame 类, LXFishStage 类, LXFishEffectGroup 类, LXFish 类, LXFishKind，LXBubble 类等。
LXFishGame 类控制游戏的整个流程,包含鱼的产生鱼的移动，游戏特效,关卡等。
LXFishStage 关卡类。
LXFishEffectGroup 特效组封装类,包含可吃鱼的图像的绘制,分数生命等信息的绘制。
LXBubble 水泡类,控制水下水泡的产生移动变化等。
LXFish 鱼类,控制鱼的缩放大小,位置,移动方向状态等鱼的信息的描述。

示例四

Hello I’m a Tree
一个简单的 3D 应用示例

其它示例

Samples/Games/2DGames/Block 打砖块游戏
Samples/Games/2DGames/Chess 中国象棋网络对战
Samples/Games/2DGames/Fishing 大鱼吃小鱼
Samples/Games/2DGames/Look 连连看
Samples/Games/2DGames/Plane 雷电
Samples/Games/2DGames/SpiderCard 蜘蛛纸牌
Samples/Games/2DGames/Tank 坦克大战
Samples/Games/2DGames/zuma 祖玛

Samples/Games/3DGames/